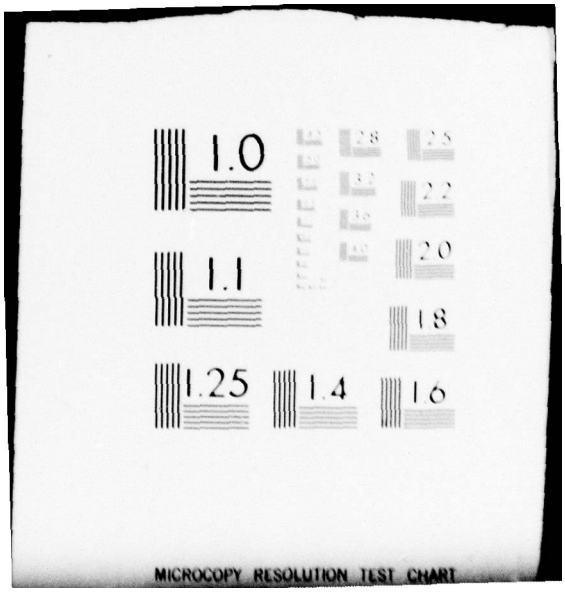
AD-A074 157 OREGON STATE UNIV CORVALLIS DEPT OF ELECTRICAL AND C--ETC F/G 17/9
SIMULATION OF COMMUNICATIONS PROTOCOLS FOR A TACTICAL RADAR NET--ETC(U)
AUG 79 C J WARNER, J D SPRAGINS AFOSR-78-3619

UNCLASSIFIED AFOSR-TR-79-0985 NL

| OF |
AD A074157

END
DATE FILMED
10-79
DDC


1.0
1.1
1.25
1.4
1.6
2.0
2.2
2.5
1.8
MICROCOPY RESOLUTION TEST CHART

SIMULATION OF COMMUNICATIONS PROTOCOLS FOR A TACTICAL RADAR NETWORK

C.J. Warner and J.D. Spragins
Electrical and Computer Engineering
Oregon State University
Corvallis, Oregon 97331

AFOSR-TR- 79-0985

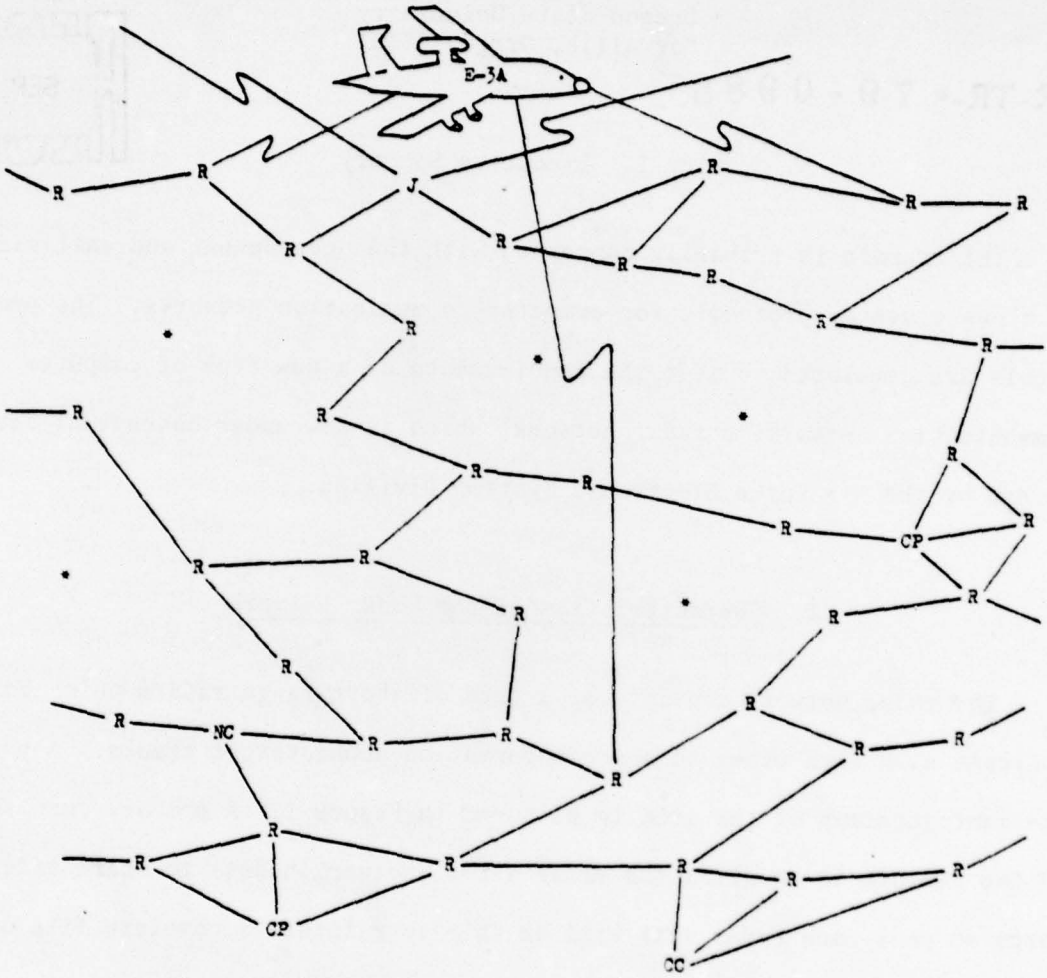
Part 1. Executive Summary

This report is primarily concerned with the development and analysis of a new class of protocols for computer communication networks. The protocols are developed to meet the requirements of a new type of computer communication network, a radar network, which is now under conceptual development by the Air Force Electronic Systems Division.

1. Characteristics of the Radar Network

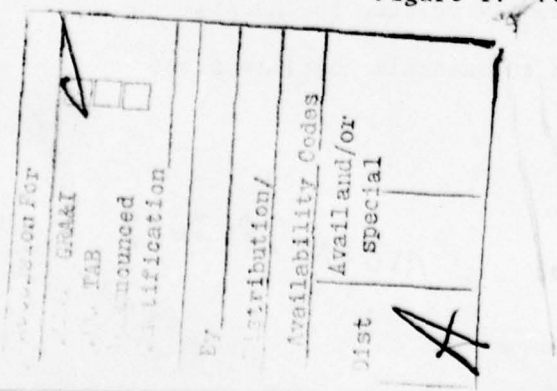
The radar network consists of a grid of short-range radars which communicate with each other to share information about target tracks. A possible configuration of the grid is pictured in Figure 1. A primary purpose of the network is to allow the radar sites (network nodes) to share track reports so that each radar site will be able to maintain a complete file of all target tracks generated anywhere within the network.

In addition to the radar sites, the network will contain command and control and other specialized nodes. These network nodes will transmit and receive specialized messages, using the radar network as their communication medium, even though their messages may not be directly related to the primary network function of allowing the radar sites to maintain complete files of target tracks.



R - RADAR NODE
* - DISABLED RADAR
J - JTIDS DISTRIBUTION STATION
NC - NETWORK CONTROL CENTER
CP - COMMAND POST
CC - COMMAND/CONTROL CENTER
E-3A - AIRBORNE COMMAND CENTER

Figure 1. Possible form of radar network.



(19) REPORT DOCUMENTATION PAGE

BEFORE COMPLETING FORM

1. REPORT NUMBER: AFOSR-TR-79-0985	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): Simulation of Communications Protocols for a Tactical Radar Network. Part I. Executive Summary.	5. TYPE OF REPORT & PERIOD COVERED: Final rept.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): C. J. Warner and J. D. Spragins	8. CONTRACT OR GRANT NUMBER(s): AFOSR-78-3619
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Oregon State University, Department of Electrical & Computer Engineering, Corvallis, OR 97331	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F 2304/A6
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research/NM, Bolling AFB, Washington, DC 20332	12. REPORT DATE: August 1979
	13. NUMBER OF PAGES: 85
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (12) 80 p.	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Radar, Network, Tactical Air Control, Data Processing, Data Communications, Computer Networks, Command and Control, Computer Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An advanced forward area tactical radar network is now under conceptual development by the Air Force Electronic Systems Division. The proposed network will link together a number of short-range radars, each with associated data processing equipment, so that each radar site has a complete file of tracks for all targets seen by any radar in the network. In addition, the communication network is expected to be used for transmission of a variety of other types of command and control messages.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

410 595

20. Abstract (continued)

A new class of adaptive routing protocols for computer communication networks has been developed in this research using the radar network as a basis. These new routing protocols utilize new techniques for searching out and using both reciprocal paths (paths over which messages can travel in either direction) and non-reciprocal paths (paths over which messages can travel in only one direction) for directed message routing.

A GPSS simulation model of a thirteen node radar network was used to determine the transient characteristics of the new routing protocols while adapting to various cases of network damage. These tests indicate that the new routing algorithms possess varying degrees of adaptability to network damage. Some of the new routing algorithms were shown to possess the capability to adapt to extreme cases of network damage. Further transient tests indicated that when some of the new routing algorithms are combined with acknowledgement techniques, complete protocols which reliably deliver all messages to their destinations, even following severe network damage, are formed.

Part 1 of the report is an "Executive Summary." Part 2, "Detailed Analyses," is essentially identical to C.J. Warner's Oregon State University Ph.D. Dissertation, "Analysis of New Protocols for Computer Communication Networks."

Each radar site will contain host computers which will, upon sighting a target, employ sophisticated data processing techniques to generate a target track. Each radar site will be capable of utilizing the tracks it generates from its own sightings, together with the target track reports it receives through the communication network, to update its own track file and to generate new track reports.

A subset of the radars seeing any given target will form a local group which will be in charge of tracking that particular target. Members of the local group share complete information on their detections of their particular target. When any member of the local group determines that it has information indicating a new track report is needed, it generates one and disseminates it throughout the network.

The track reports transmitted by the local group members, which are sent to all other members of the network, are by far the most common members of a class of transmissions which will be called nondirected messages, since they are not really directed toward any particular node, and are expected to disseminate the same information to all nodes.

Some of the track reports sent out by the members of each local group, as well as the messages sent out by the command and control and other specialized nodes, will probably be directed messages sent from one particular node to another. (To ensure the proper operation of the communication protocols developed as a part of this research, the specialized nodes may be required to occasionally transmit non-directed messages.)

The types of communication links which will be used between nodes are likely to be microwave radio links using narrow-beam antennas. Each communication link would have separate transmitting and receiving antennas. This

would provide for bi-directional simplex transmission of messages between nodes; i.e., the send and receive channels would operate independently of one another. This implies that the two directions of transmission can be treated independently, since it is entirely possible only one direction will be operational.

The basic switching technique used will be a form of packet switching. Target tracks will be transmitted over the bi-directional simplex communication channels in fixed size data blocks, probably approximately 300 bits in length. Other miscellaneous types of specialized transmissions may use varying sizes of data blocks, although there is some possibility of fitting most, if not all, of these into the same size packets.

2. A GPSS Simulation Model of The Radar Network

A GPSS simulation model of the radar network has been written in order to evaluate the adaptive routing and acknowledgement protocols developed. The program models a thirteen node network, as shown in Figure 2. Each node in Figure 2 has been assigned an identifying number. Also, each direction of each bi-directional link has been assigned a unique channel identification number.

Each node in the thirteen node GPSS network generates both directed and non-directed messages. The inter-generation times for these two types of messages are exponentially distributed, with per node means of $\bar{t}_D$ and $\bar{t}_{ND}$ respectively. Both the directed and non-directed messages are generated with their source nodes uniformly distributed throughout the network. The destinations for the directed messages are also uniformly distributed throughout the network.

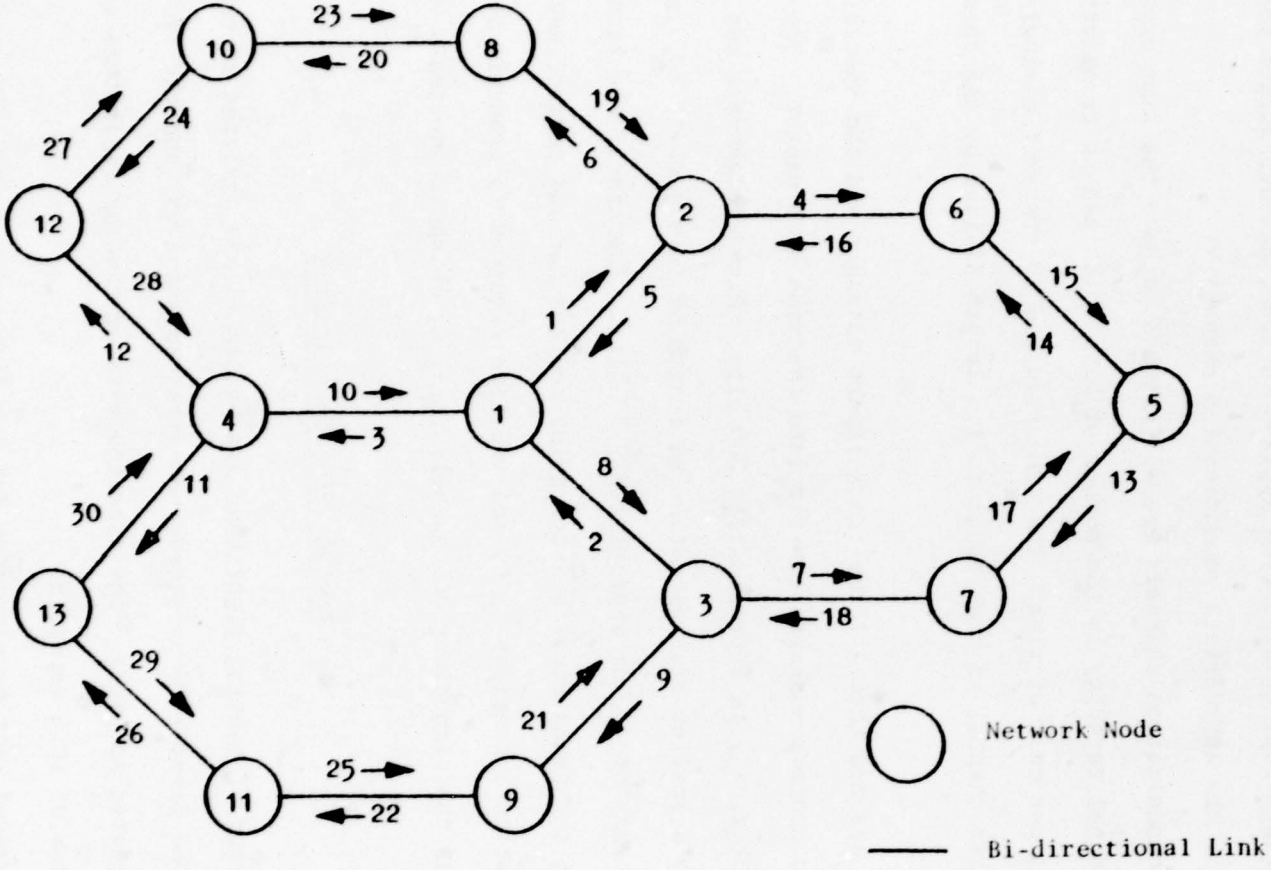


Figure 2. Thirteen node GPSS network.

Each node receives over the communication lines messages from each of its neighbors which are processed along with the messages the node generates itself. These messages are handled in the order in which they arrive, have the appropriate routing algorithm applied to them, and are queued for transmission over the appropriate communication channel.

The communication channels are all assumed to have the same capacity, C. The channel capacity is the number of bits which can be transmitted over the channel per unit of time. The time required to transmit a single message over a single channel is thus equal to its length divided by the channel capacity.

It is assumed that all the track report messages and the specialized command and control messages can fit into the same size packet. The length of this packet, ℓ_M, is approximately 300 bits. Acknowledgements are assumed to fit into a smaller size packet. The length of this packet, ℓ_A, is assumed to be one tenth of ℓ_M, or approximately 30 bits. Therefore, the time required to transmit the track reports and specialized command and control messages over a communication channel is ℓ_M/C, and the time required to transmit acknowledgements across the communication channels is ℓ_A/C, which is one tenth ℓ_M/C.

3. Routing Protocols Studied

Two types of routing algorithms are presented and analyzed in this section. The first is for routing non-directed messages, and the second for routing directed messages. The directed message routing algorithms are the primary focus of this research.

The favored routing algorithm for non-directed messages is a form of flooding whereby each node which receives a non-directed message relays the

transmission on to all neighboring nodes except the one from which the message came. It is easy to verify that this algorithm will quickly disseminate such messages to all nodes to which communication paths exist.

By the nature of the algorithm, flooding adapts instantly to network damage, communication link jamming, node additions and deletions, and other changes in network structure. Flooding requires no knowledge of the network structure and requires essentially no processing. It requires some increase in traffic over that required by so-called "optimum" algorithms (at most 100% more traffic but probably considerably less), but the advantages listed appear to outweigh this consideration.

A new class of directed message routing algorithms have been developed to meet the requirements of the radar network. These new algorithms route messages by storing at each node a table of the current estimate of the time required to reach each possible destination when leaving via each output port plus possibly some other information. The table and other stored data are consulted for each message arriving at the node, to determine which output port is the best to route the message to its destination. The message is then sent out of this port. With these new routing algorithms, the routing tables are updated with information contained in the incoming non-directed messages. Since the non-directed messages are routed using flooding, which has excellent adaptability, it is expected that this adaptability can be reflected in the routing tables, making a very reliable algorithm.

Several versions of the new routing algorithms have been developed, each with its own performance characteristics. The concepts behind the operation of each routing algorithm are described below. The performance characteristics of each algorithm are in section 4.

3.1 Simple Backwards Learning

With Simple Backwards Learning (SBL), each node stores a routing table, as shown in Figure 3. Each routing table entry, a_{ij}, is roughly proportional to the estimated delay when sending a directed message to node j by way of port i. Thus, each column in the routing table corresponds to one of the network nodes and each row corresponds to one of the outgoing ports. For the simulation model of the network, each node thus stores a routing table with thirteen columns and two or three rows.

Each column corresponds to one node.

Each row corresponds to one of the node's ports.

a_{ij} = a rough estimate of the delay required to reach node j when the message is sent out of port i.

Figure 3. SBL routing table.

The routing table entries are updated using information contained in the flooded messages. For this purpose, each flooded message contains, in addition to the message content, the source node address, S, and an estimate, D_{MSG}, of the time expired in reaching the current node from node S.

Suppose that a flooded message with source node S arrives at node X by way of port P. Node X first checks to see if the received copy is the first copy of the message. If it is, the node updates its routing table entry for port P as follows:

$$a'_{PS} = a_{PS} + k_1(D_{MSG} - a_{PS}) \qquad (1)$$

where

a'_{PS} = the new estimated delay to node S by way of port P.

a_{PS} = the old estimated delay to node S by way of port P.

k_1 = the learning constant ($0 < k_1 \leq 1$).

D_{MSG} = the delay experienced by the message in traveling from node S to node X.

Thus, the new delay estimate, a'_{PS}, is a fraction k_1 of the way between the old estimate, a_{PS}, and the current estimate, D_{MSG}. Immediately after updating the entry for port P, the entries for the other ports are increased, thus causing them to be "forgotten". This is an essential step to ensure that old routing table entries which are no longer valid because of changes in network structure do not remain misleadingly small.

After the node has updated its routing tables, it sends the non-directed message on to each of its neighbors, except the one from which it was received, completing the flooding process for that message at node X.

If a second copy of the non-directed message arrives, the node updates the entry for the port the copy arrived in, using Equation (1). This second copy does not, however, cause the other routing table entries to forget their values since the forgetting process was already accomplished by the first copy of the message. After updating the routing table, the message is then discarded, since the neighbors have already been sent a copy of it.

Now suppose that a directed message arrives at node X, and is directed for node D. Node X first checks to see if a copy of the directed message had ever visited the node previously. If it has, the message is discarded and (in the simulation runs) a global counter of looping messages is incremented. If it is the first copy of the message, the routing tables are

consulted to determine the best port(s) to send the message out of to reach node D. A message is sent out of port M if

$$(((a_{MD} \leq a_{iD}) \text{ or } (\text{link out of port } i \text{ is down})) \underline{V}\ i)$$
$$\text{and } (\text{link out of port M is up}) \qquad (2)$$

3.2 Reciprocal Path Search Algorithm

There is a basic problem with the Simple Backwards Learning algorithm in that it assumes reciprocity of the message paths. That is, it assumes that if messages can travel in one direction over a path, they can travel in the opposite direction also. This is a false assumption when links are jammed, and causes the algorithm to perform poorly when the network is subjected to jamming. In order to remedy this problem, information concerning non-reciprocal paths must be sent to all nodes in the network, and the nodes must use this information to construct better routing tables. The problem, then, is twofold. First, pertinent jamming information must be communicated to the network nodes. Second, adaptation techniques using this information must be developed.

The Reciprocal Path Search (RPS) algorithm solves the first problem by attaching to each non-directed message a "Non-Reciprocal Path Indicator" (NRPI) bit. This bit is initially set to zero when the message is originated at its source node. Then, whenever the non-directed message traverses a link which is non-reciprocal, i.e., being jammed, the NRPI is set to one.

The nodes use the information contained in the NRPI to flood out extra non-directed messages to search out the reciprocal paths, which can then be learned by the routing tables. Only the flooded messages which arrive over

reciprocal paths are used to update the tables so that the tables will only learn the reciprocal routes. Thus, the routing tables send directed messages over routes which contain only reciprocal paths, avoiding jammed links. This allows the algorithm to perform well even in the presence of jamming as long as a reciprocal path does exist between all pairs of nodes. This algorithm does not adapt well to severe cases of jamming in which there are source-destination node pairs which are connected only by non-reciprocal paths (i.e., paths which contain jammed links).

3.3. Rapid Reciprocal Path Search Algorithm

The Rapid Reciprocal Path Search (RRPS) algorithm is designed to adapt to cases of jamming more quickly and completely than does the Reciprocal Path Search algorithm. It accomplishes this by using the NRPI bit attached to the non-directed messages to determine when a link has suddenly become non-reciprocal, and allow the algorithm to utilize alternative routing techniques to temporarily circumvent the normal process of learning and forgetting the routing table entries. This results in an algorithm which adapts very quickly.

This algorithm is also designed to sense when no reciprocal path exists between nodes, and thus it is able to take appropriate measures to ensure that the message is delivered to its destination even in extreme cases of network jamming.

The Rapid Reciprocal Path Search algorithm uses three routing tables, as shown in Figure 4. The Primary Routing table (A table), the Selection Table (B table), and the Secondary Routing Table (C table).

Primary Routing Table (A table)

When in use

a_{ij} = a rough estimate of the delay required to reach node j when port i is used.

Selection Table (B table)

If $b_{ij} = 0$, use a_{ij} for delay estimates.

If $b_{ij} > 0$, use c_{ij} for delay estimates.

Secondary Routing Table (C table)

When in use

c_{ij} = a very rough estimate of the delay required to reach node j when port i is used. All entries updated only by messages using reciprocal paths.

Figure 4. RRPS Algorithm Routing Tables.

The Primary Routing Table is used mostly when no network damage exists, or the damage is only reciprocal in nature. It is updated by the first copy of a message, even when that copy comes in over a non-reciprocal path. The Selection Table is updated using the NRPI bits. A Selection Table entry of $b_{ij} = 0$ indicates that the a_{ij} entry was last updated by a message which had an NRPI = 0 (i.e., the message came in over a reciprocal path), and is thus useful for routing directed messages. A value of $b_{ij} = 1$ indicates that the a_{ij} entry was last updated by a message which arrived with an NRPI = 1, and thus signals immediately (circumventing the forgetting process) that the a_{ij} is no longer useful for routing directed messages. When $b_{ij} = 1$, the algorithm uses the Secondary Routing Table entry, c_{ij}, for routing directed messages. The Secondary Routing Table entries are updated only by the extra flooded messages which are generated to search out the reciprocal paths. For any destination node, the algorithm is designed so that the Secondary Routing Table entries are all equal to each other as well as to the Primary Routing Table entries which correspond to previously unused ports, at the time of network damage. This allows the initial updates of the table entries (by extra flooded messages which search out reciprocal paths) to quickly indicate which port to take to use this reciprocal path. If no reciprocal path exists, the fact that all the routing table entries chosen by the Selection Table are equal indicates that no reciprocal path exists, allowing the node to take appropriate action to ensure the message delivery. In this case, the algorithm will flood the directed messages out at the nodes where this condition exists. This flooding is local only, however, since other nodes receiving copies of the flooded message route them by the same directed message routing algorithm, which involves flooding only when no reciprocal path exists.

3.4 Reciprocal Path Search Algorithm with Delay Vectors

Each of the previous algorithms was only designed to search out and use reciprocal paths, i.e., paths over which messages can travel in both directions. The Reciprocal Path Search Algorithm with Delay Vectors (RPSDV) is designed not only to search out and use the reciprocal paths, but also to search out non-reciprocal paths and use them for routing directed messages. This involves locating non-reciprocal (i.e., jammed) links which can be used for routing directed messages, and disseminating information on these links to all the network nodes so that they can decide if the links are useful to them for routing purposes. In this way, each node will have information on a unique path to a destination node whenever such a path exists, reciprocal or non-reciprocal.

For the purpose of storing information on both the reciprocal and non-reciprocal paths, each node will maintain two tables, the A table and the Delay Vector Table. The A table is the same routing table as that used previously by the Reciprocal Path Search algorithm and stores information on reciprocal paths. It is updated by flooded messages arriving over reciprocal paths. The Delay Vector Table is a new table and stores information on non-reciprocal paths. It is updated by special non-directed messages called "delay vectors." These delay vectors disseminate information on particular non-reciprocal links throughout the network, providing estimates of the delay required to reach destination nodes via the non-reciprocal link, as well as information defining the particular non-reciprocal link in question.

A typical Delay Vector Table is shown in Figure 5. It has a column for each destination node, and three rows. The first row's entry, d_{1n}, contains the delay estimate to node n via the non-reciprocal link defined by the d_{2n}

d_{1n} = The delay estimate to node n when the directed message is routed via the link connecting the intermediate and creation nodes.

d_{2n} = The intermediate node.

d_{3n} = The creation node.

Figure 5. Delay Vector Table.

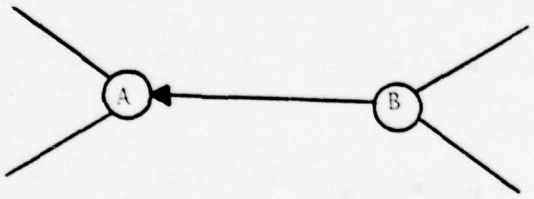


Figure 6. Non-reciprocal link.

and d_{3n} entries. To understand the second and third row entries, consider Figure 6 which shows a non-reciprocal link connecting nodes A and B. Node B is called the intermediate node (since it is intermediate along any path which includes the non-reciprocal link), and node A is called the creation node (since it creates the delay vectors which disseminate information on the non-reciprocal link). The non-reciprocal link is defined as the link connecting nodes A and B, with the direction of transmission in the direction from the intermediate node, B, toward the creation node, A. Delay Vector Table entry d_{2n} is used to store the identity of the intermediate node, and entry d_{3n} is used to store the identity of the creation node.

Note that the Delay Vector Table only stores a delay estimate to a destination node over a particular non-reciprocal link, along with the identity of the nodes which the link connects. In order to be able to use this information, the node must be able to determine how to reach node B, from whence a message can be sent across the link to node A and on to its destination. The only table which the node can use to route messages to node B is the A table. Thus, in order to use the Delay Vector Table entries, a node must use the A table to route the message to node B, and from there over the link and ultimately to its destination.

When routing a directed message, a node consults its A table and its Delay Vector Table to determine if the message should be routed to its destination node directly over a reciprocal path (if one exists) or if it should be routed to its destination over a non-reciprocal path (if one exists). If neither a reciprocal path nor a non-reciprocal path exists to the destination node, meaning that the node at which the message currently resides is completely disconnected from the destination node, the algorithm senses this

and simply does not send the message out. None of the previous routing algorithms possess this capability to sense when nodes are disconnected from each other.

3.5 Rapid Reciprocal Path Search Algorithm with Delay Vectors

This algorithm combines the Rapid Reciprocal Path Search algorithm with the use of delay vectors to create an algorithm which adapts very quickly to network damage, yet has the capability to use non-reciprocal links for routing directed messages.

The Rapid Reciprocal Path Search Algorithm with Delay Vectors (RRPSDV) stores routing information in four tables. The A, B, and C tables contain information on reciprocal paths while the Delay Vector Table stores information on non-reciprocal paths. The techniques for creating delay vectors and using them to update the Delay Vector Tables are exactly the same as in the Reciprocal Path Search Algorithm with Delay Vectors.

When routing a directed message, a node consults its A, B, and C routing tables to determine if the message should be routed over reciprocal paths, and it consults its Delay Vector Table to determine if the message should be routed over a non-reciprocal path. Preliminary simulation runs indicated that unless some restrictions are placed upon the use of the Delay Vector Table entries for routing purposes, an excessive amount of looping occurs, eliminating the quick adaptability which is a prime goal of this algorithm. The looping problem necessitated that the nodes be allowed to use the Delay Vector Table entries for routing purposes only when using them provides a clear benefit. Therefore, nodes are prevented from using the Delay Vector Table entries, even though they indicate a shorter path exists

over a non-reciprocal link, when using them results in sending the message out of the same port in which it arrived. In this case, the node falls back on the use of the A, B, and C tables. This restriction makes clear sense, since allowing a message to be transmitted out of the port in which it came will result in ping-ponging, and the loss of the message. The preliminary simulation runs also indicated that a second modification of preventing a source node for a directed message from using its Delay Vector Table for routing directed messages further reduced ping-ponging.

If neither a reciprocal path nor a non-reciprocal path exists to the destination node, the algorithm senses this and simply does not send the message out.

The use of the A, B, and C tables for routing messages over reciprocal paths allows the nodes to use the same techniques as did the RRPS algorithm to very quickly determine when a link has suddenly become non-reciprocal (probably because of jamming), and thus adapt quickly to this type of network damage.

4. Adaptability Tests of the New Routing Protocols

Transient simulation tests were conducted on each of the new routing algorithms to determine their performance characteristics while adapting to three cases of network damage. The first case of network damage has three of the communication links destroyed so that transmission is inhibited in both directions over them. This is a form of reciprocal damage, since the damage is the same in both directions over the links. This case is shown in Figure 7. The destroyed links are omitted from the diagram. Figure 8 shows the second case of network damage. Here the same three links are jammed

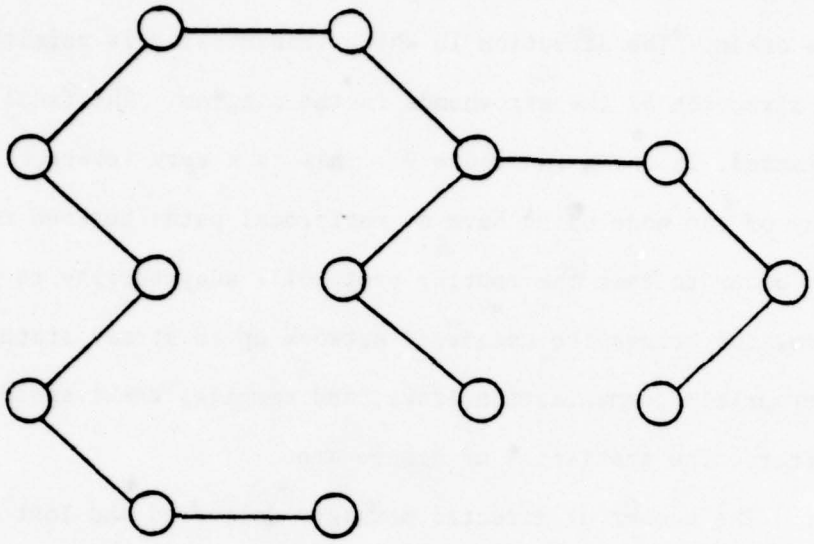


Figure 7. Three links destroyed.

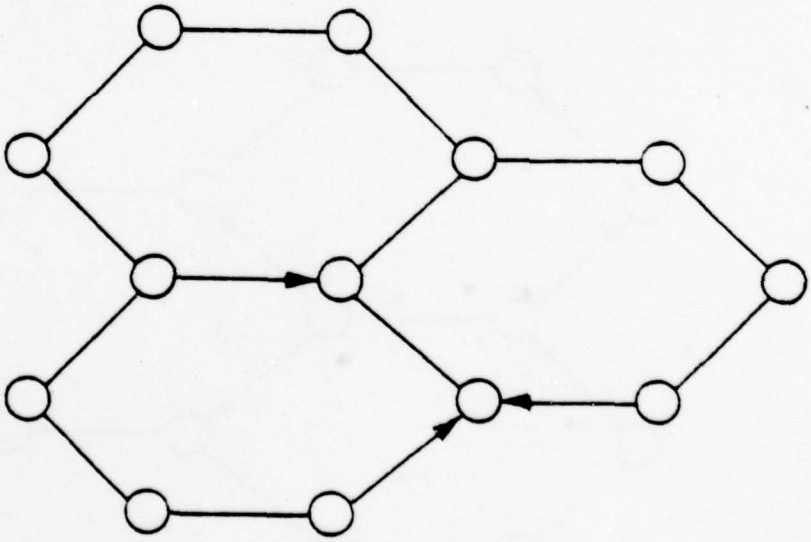


Figure 8. Three links jammed.

instead of destroyed. Jamming is a form of non-reciprocal damage whereby transmission is permitted in one direction over a communication link but not the other. The direction in which transmission is permitted corresponds to the direction of the arrowheads in the diagram. The final case, six links jammed, is shown in Figure 9. This is a very severe case of jamming, and many of the node pairs have no reciprocal paths between them.

In order to test the routing protocol's adaptability to network damage, the simulator brings the undamaged network up to steady state, disables the appropriate communication links, and compiles the transient statistics thereafter. The statistics used here are:

1. The number of directed messages delivered and lost per directed message sent.
2. The number of looping messages per directed message sent.
3. The adjusted directed message delay and non-directed message delay.

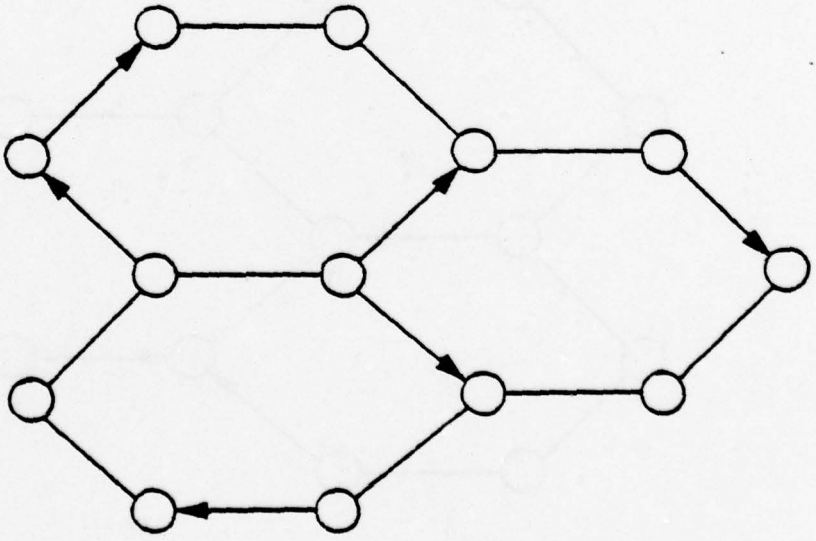


Figure 9. Six links jammed.

These statistics are averaged over a "window" of width $4\bar{t}_{ND}$ for statistics 1 and 2 above, and of width $8\bar{t}_{ND}$ for 3. Thus, any point of the curves of these statistics provides an indication of the near term performance of the routing protocols.

Some explanation should be made concerning the statistic concerning message delays. The average directed message delay is basically computed by averaging the elapsed time between when the directed message is originally transmitted at its source node and when it is successfully received at its destination node. This is an average over all directed message transmissions. A weakness of the average directed message delay statistic computed in this manner is the fact that only the messages which reach their destinations are averaged in. Lost messages are not included in the average. The simulation results have shown that following network damage this average is weighted in favor of those source-destination pairs which are close together, resulting in a misleadingly small value for the average delay. In order to compensate for this, a penalty for each lost message has been included in the averages shown here. The size of the penalty is chosen to be 24 (ℓ_M/C). This new quantity, the adjusted directed message delay is often a better indicator of the delay performance of the routing algorithms. Lost messages do not cause the same type of problem, however, when acknowledgements are used with re-transmissions of messages when no acknowledgement is received. After one or several transmissions, each directed message will eventually reach its destination, and be included in the average.

The values of the various simulation and routing algorithm parameters which were used in the transient simulation tests are given in Table 1. These parameter values will remain fixed throughout the transient simulation

tests of all the routing algorithms, unless otherwise specifically stated.

Table 1. Transient Simulation Parameter Values.

Parameter	Value
k_1	0.25
k_2	0.25
k_3	10.0
$\bar{t}_D$	300.0
$\bar{t}_{ND}$	300.0
ℓ_M/C	10.0

4.1 Three Links Destroyed Case

4.1.1 Simple Backwards Learning

Figure 10 is a plot of the directed messages delivered and lost, per directed message sent, for the Simple Backwards Learning algorithm. This plot shows that immediately following the instant of network damage, a temporary condition exists when messages are lost. This condition lasts for a period of approximately $8\bar{t}_{ND}$. During this period, the routing tables have not yet learned the new routes by which flooded messages are now arriving at the nodes. Following the learning period, the simulation results show that the algorithm has completely adapted to the new network configuration, and no more messages are lost.

Figure 11 is a plot of looping messages following the instant of network damage. A message is considered to have looped if it, or a copy of it, ever returns to a node. Looping messages are another indicator of

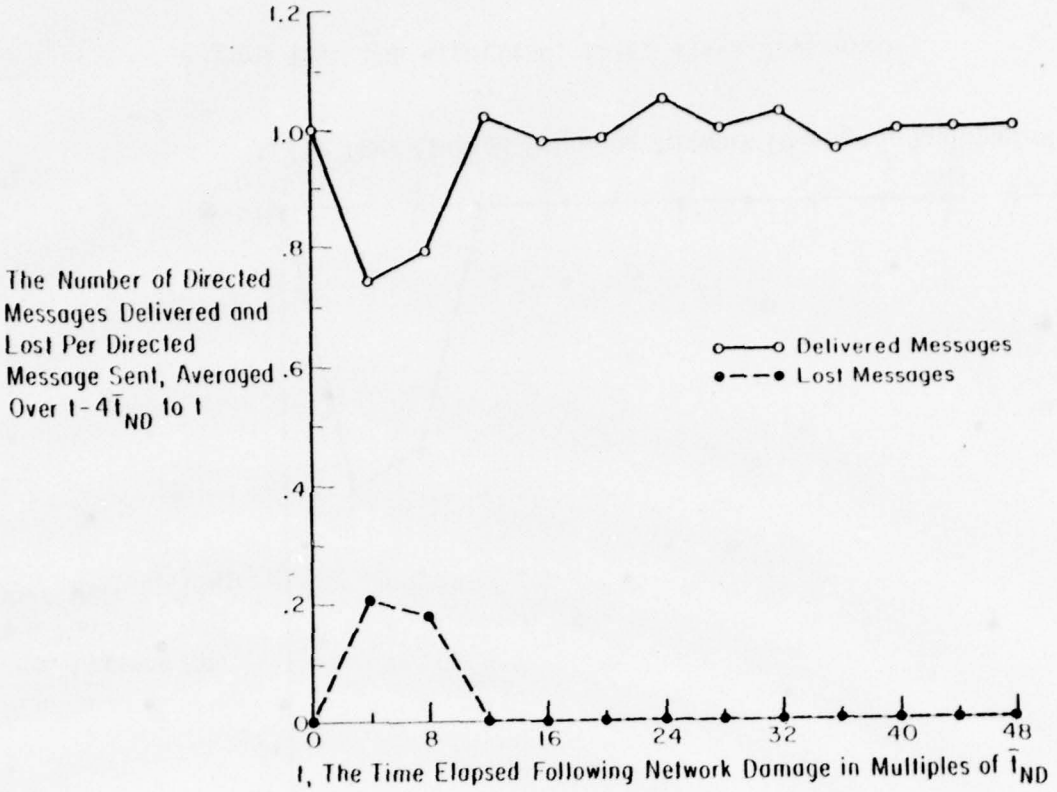


Figure 10. SBL algorithm, three links destroyed.

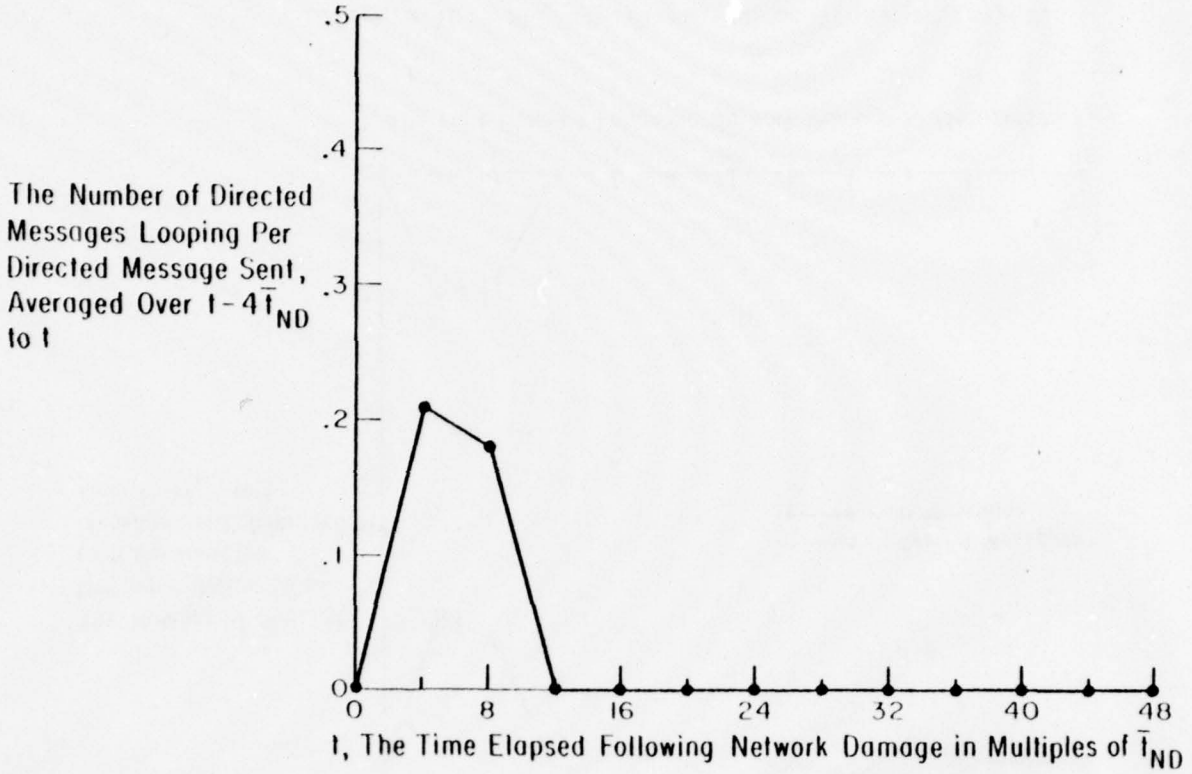


Figure 11. SBL Algorithm, three links destroyed.

bad routing table entries. As can be seen, the looping is temporary, lasting only as long as messages are being lost.

Figure 12 is a plot of the adjusted directed message delay and non-directed message delay, following the instant of network damage. Also included is a line showing the optimum delay. The optimum delay is computed by determining the average minimum path length between all node pairs, and multiplying this average by the time required for a message to traverse a single link. The result is a steady state value of the optimum delay which excludes queueing delays. A consistent increase in the directed and non-directed message delays above this optimum delay is due to network congestion and to the inability of the routing algorithms to route the messages by the shortest paths.

The non-directed messages always take the shortest path between nodes. Any difference between the non-directed message delay and the optimum delay is due to network congestion causing queueing delays. Initially, the non-directed message delay takes a jump from its pre-damage value and then settles out. The non-directed message delay remains larger than the optimum delay, indicating the existence of a certain amount of network congestion.

Any significant difference between the adjusted directed message delay and the non-directed message delay is due to either the penalty of 24 (ℓ_M/C) (which is 240 for the values given in Table 1) imposed on the adjusted directed message delay for each lost message, or due to messages reaching their destinations by longer than necessary routes. As can be seen, the adjusted directed message delay initially is much larger than the non-directed message delay. This, of course, is due primarily to the lost messages. However, once the routing algorithm has completely adapted and messages are no longer

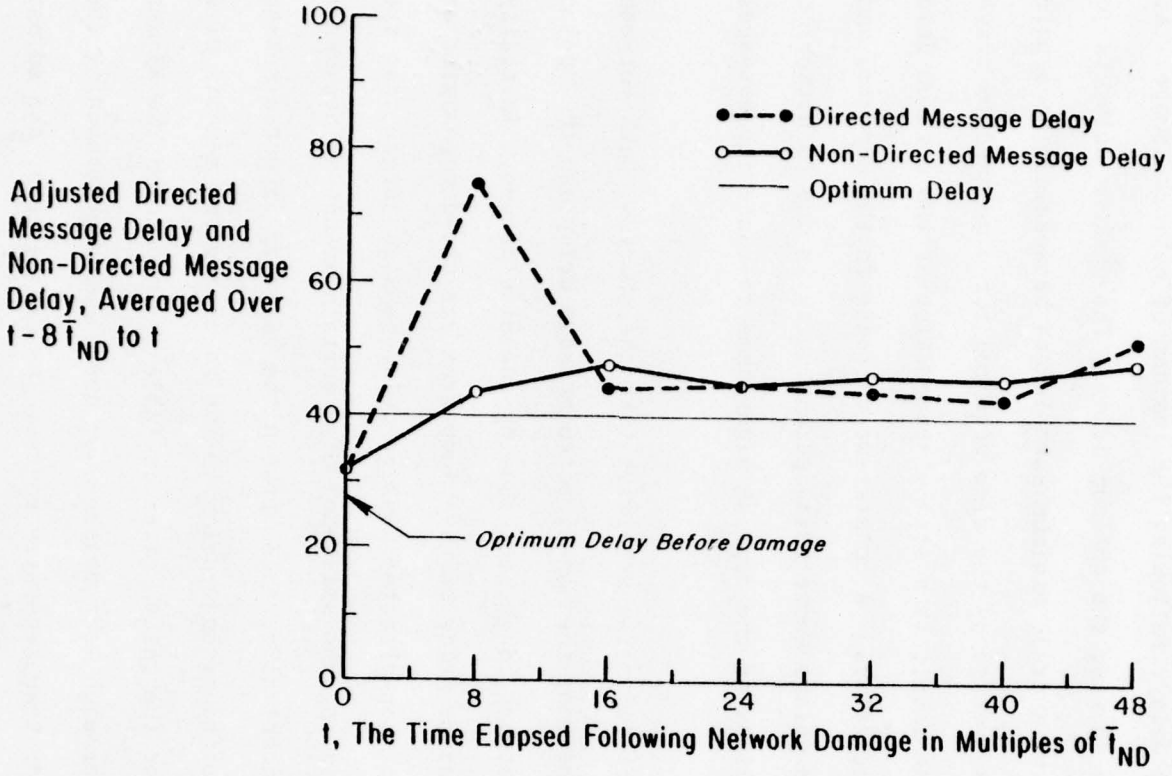


Figure 12. SBL Algorithm, three links destroyed.

being lost, the directed message delay and the non-directed message delay appear to be essentially equal, indicating that the routing tables are routing the directed messages by shortest paths.

The simulation results have shown that for the case of three links destroyed, the Simple Backwards Learning routing algorithm adapts very well. This is primarily due to the excellent adaptability of the flooding algorithm being reflected in the routing tables as the flooded messages bring in information on new paths.

4.1.2 Other Routing Algorithms

The other four routing algorithms (the RPS, RRPS, RPSDV, and RRPSDV) differ from the Simple Backwards learning algorithm primarily in how they adapt to non-reciprocal forms of damage, such as jamming. Thus, their performance characteristics while adapting to the three links destroyed case differ only slightly from the results just presented for the Simple Backwards Learning algorithm, and will not be presented here.

4.2 Three Links Jammed Case

4.2.1 Simple Backwards Learning

Figures 13 and 14 show that for the three links jammed case, messages are both lost and looping continuously following the network damage. This is a direct result of the inherent assumption by the Simple Backwards Learning algorithm of reciprocity of the communication links. That is, the algorithm assumes that if transmission is possible in one direction over a link, it is possible in the opposite direction also. Thus, the Simple Backwards Learning algorithm does not have the ability to adapt to non-

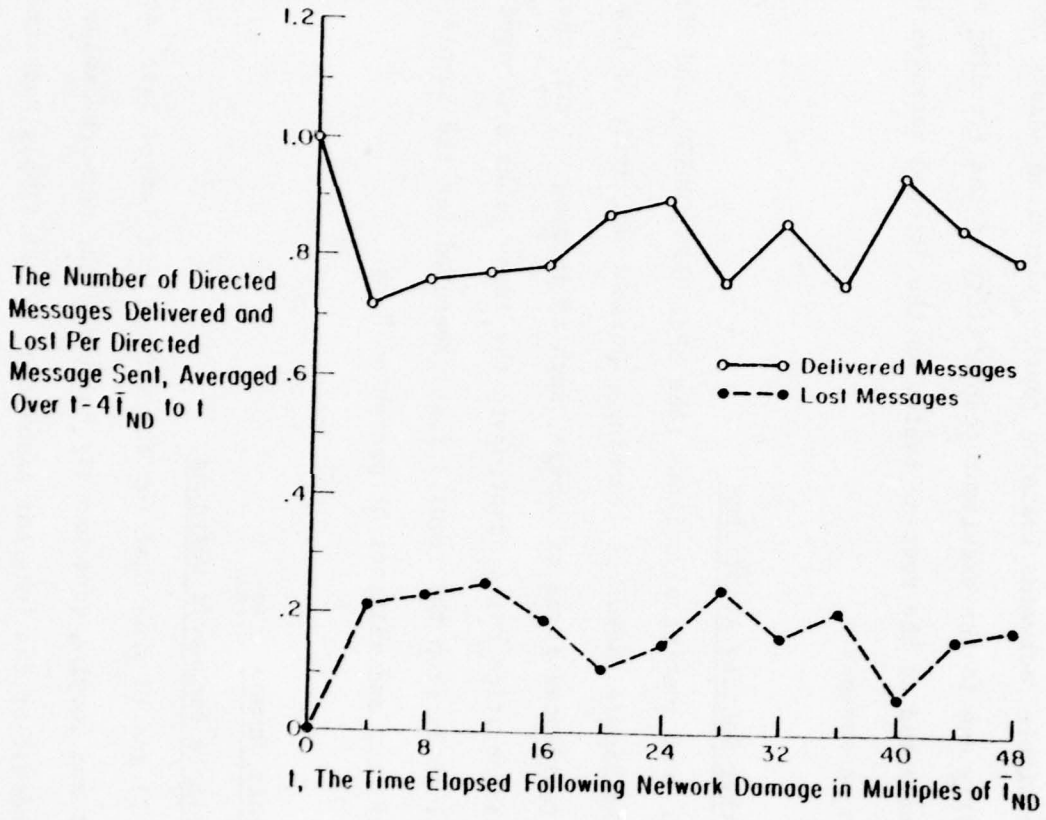


Figure 13. SBL Algorithm, three links jammed.

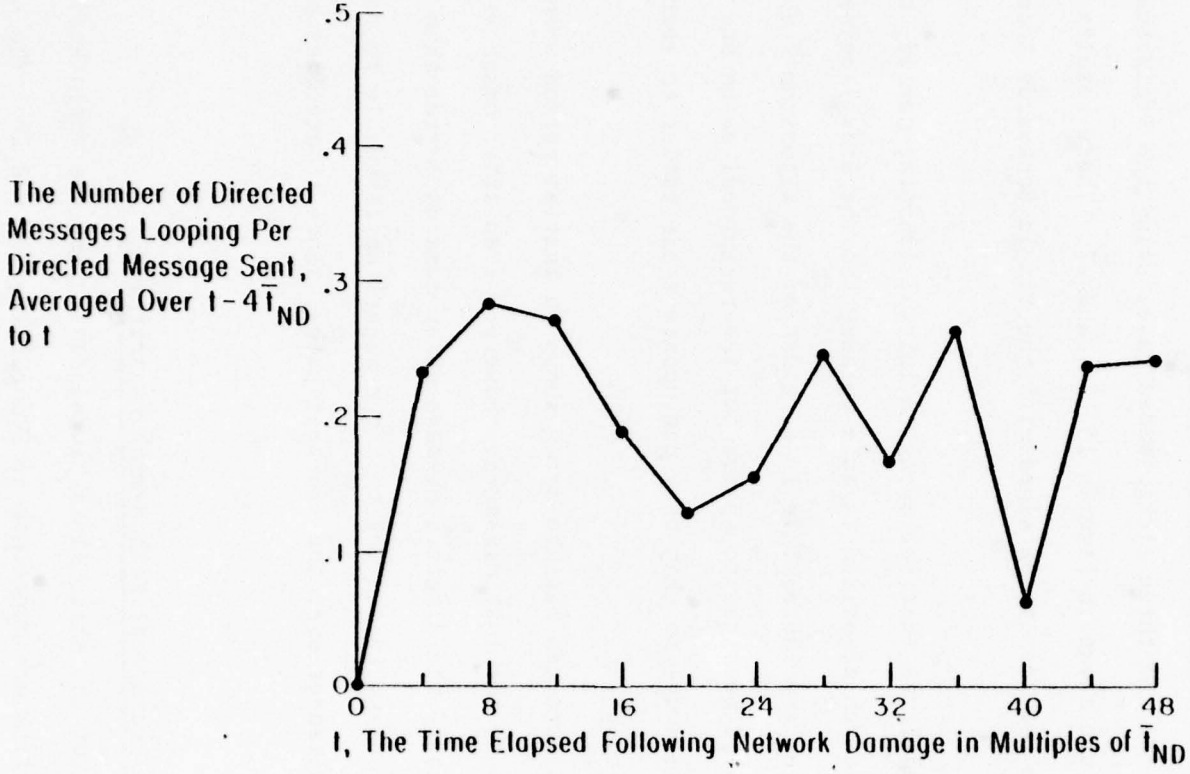


Figure 14. SBL Algorithm, three links jammed.

reciprocal forms of damage.

4.2.2 Reciprocal Path Search Algorithm

The results for the three links jammed case, using the Reciprocal Path Search algorithm are shown in Figures 15, 16, and 17. These results differ dramatically from the results obtained with the Simple Backwards Learning algorithm.

Figure 15 indicates that following an initial learning period, the Reciprocal Path Search algorithm is able to adapt to the link jamming, and no further messages are lost. This is a result of the algorithm's ability to use extra flooded messages to search out the reciprocal paths and provide information to the nodes so that they can update their tables to learn the new reciprocal paths.

The adjusted directed message delay shows an initial period when delay is very large due to the lost messages. However, even after messages are no longer being lost, the directed message delay remains larger than the non-directed message delay, indicating that the algorithm is unable to route all of the directed messages over the shortest paths, since it utilizes only reciprocal paths.

4.2.3 Rapid Reciprocal Path Search Algorithm

The results for the three links jammed case using the Rapid Reciprocal Path Search algorithm are contained in Figures 18, 19, and 20. The Rapid Reciprocal Path Search algorithm is able to dramatically reduce the numbers of lost and looping messages, when compared to the previous algorithms. Messages are lost for only a period of duration $4\bar{t}_{ND}$. In addition, only

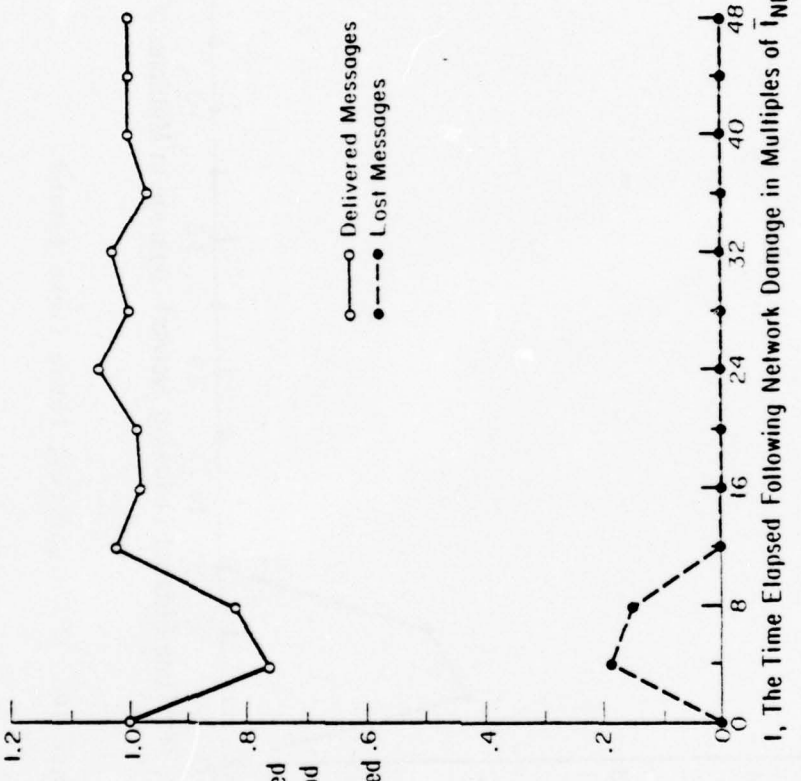


Figure 15. RPS Algorithm, three links jammed.

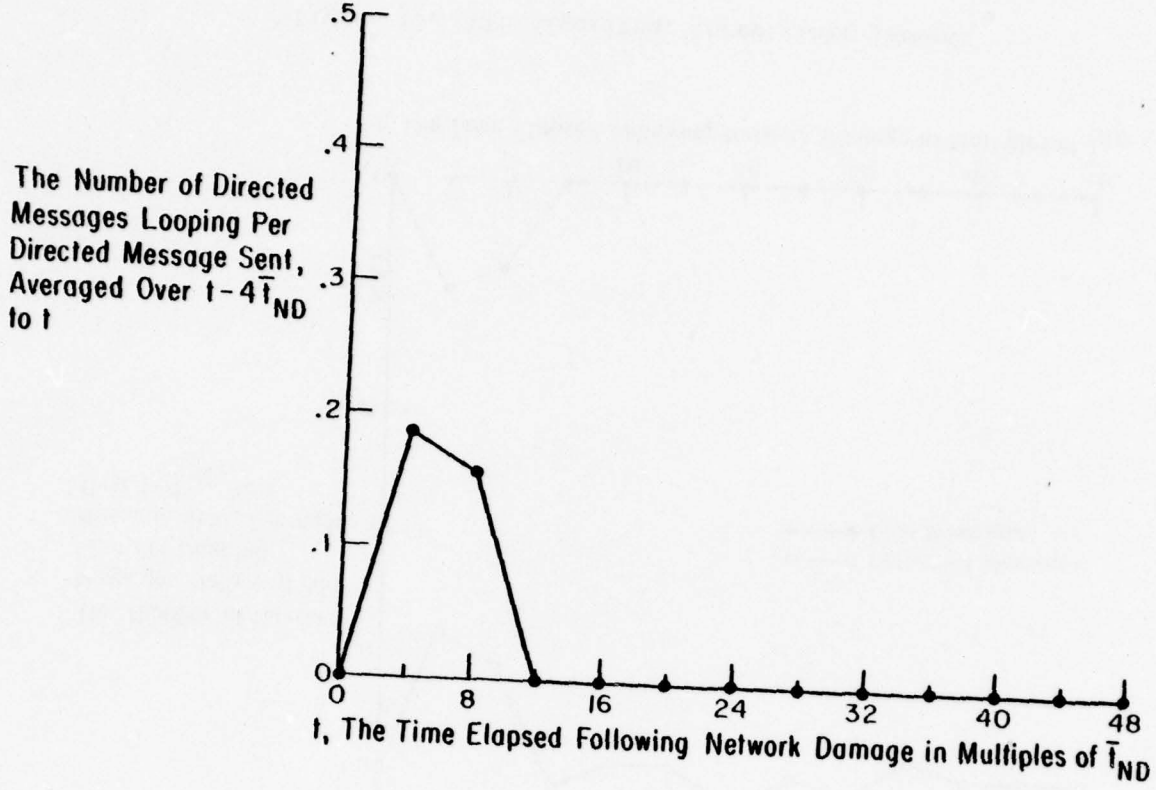


Figure 16. RPS Algorithm, three links jammed.

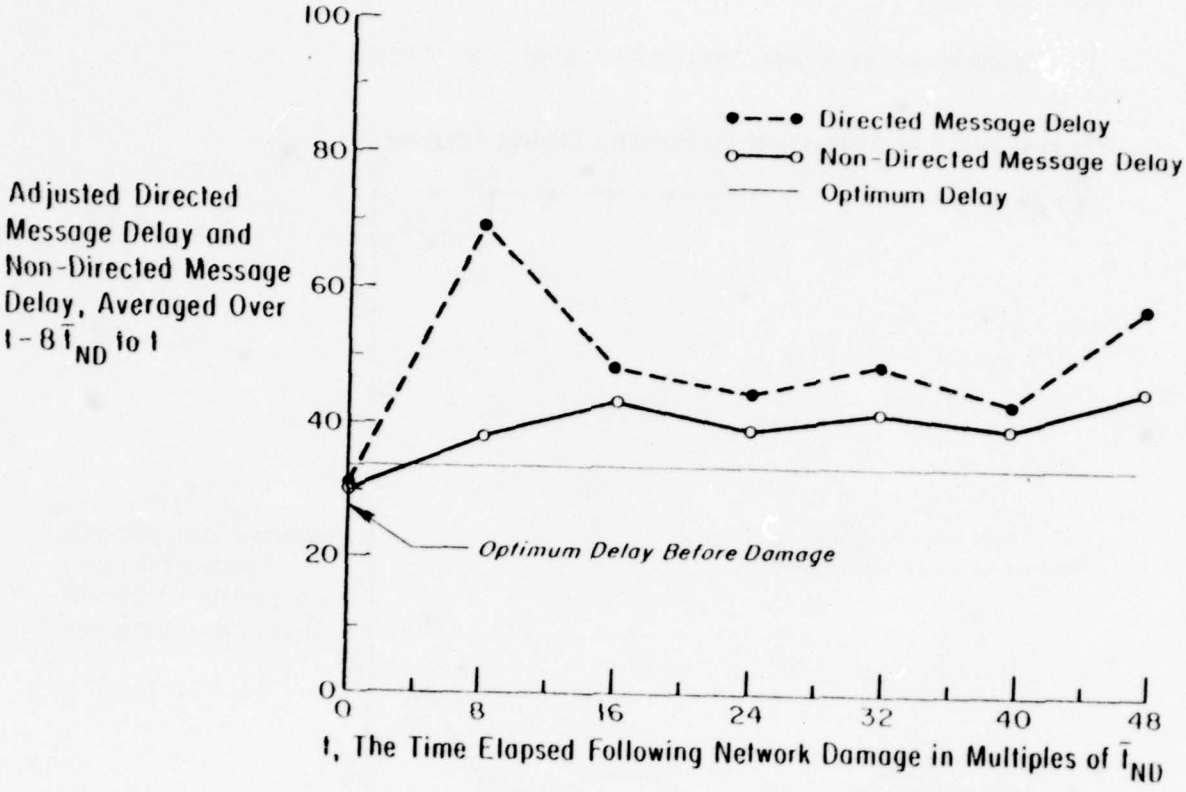


Figure 17. RPS Algorithm, three links jammed.

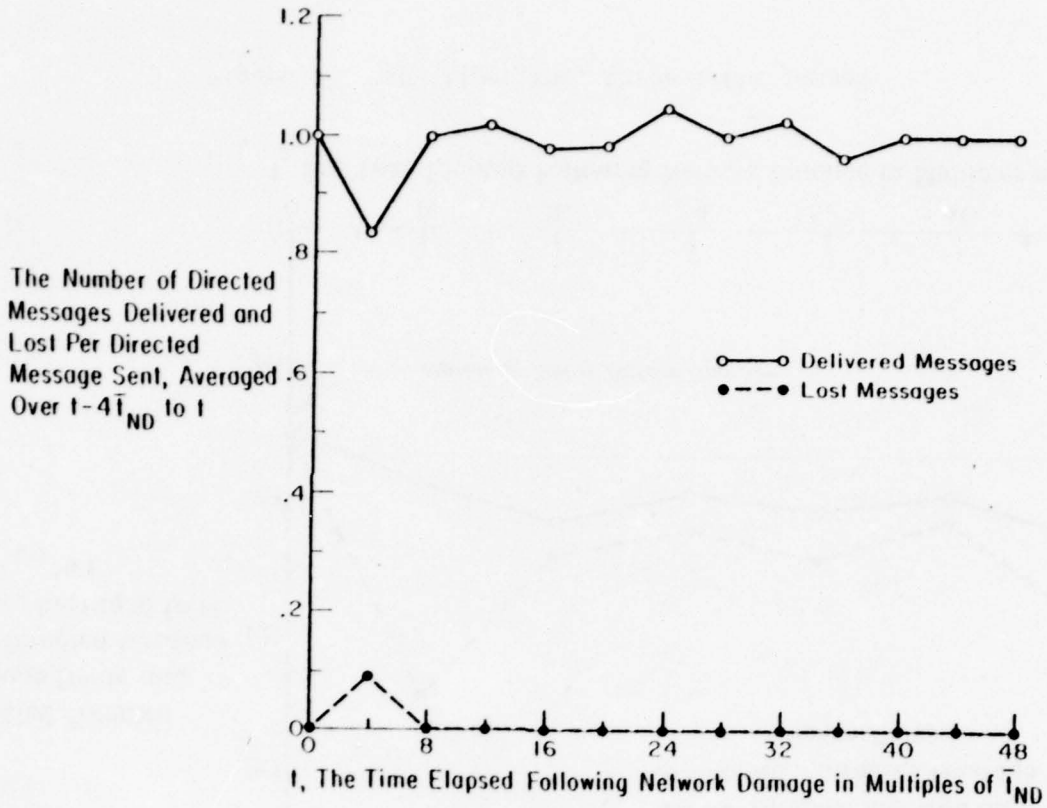


Figure 18. RRPS Algorithm, three links jammed.

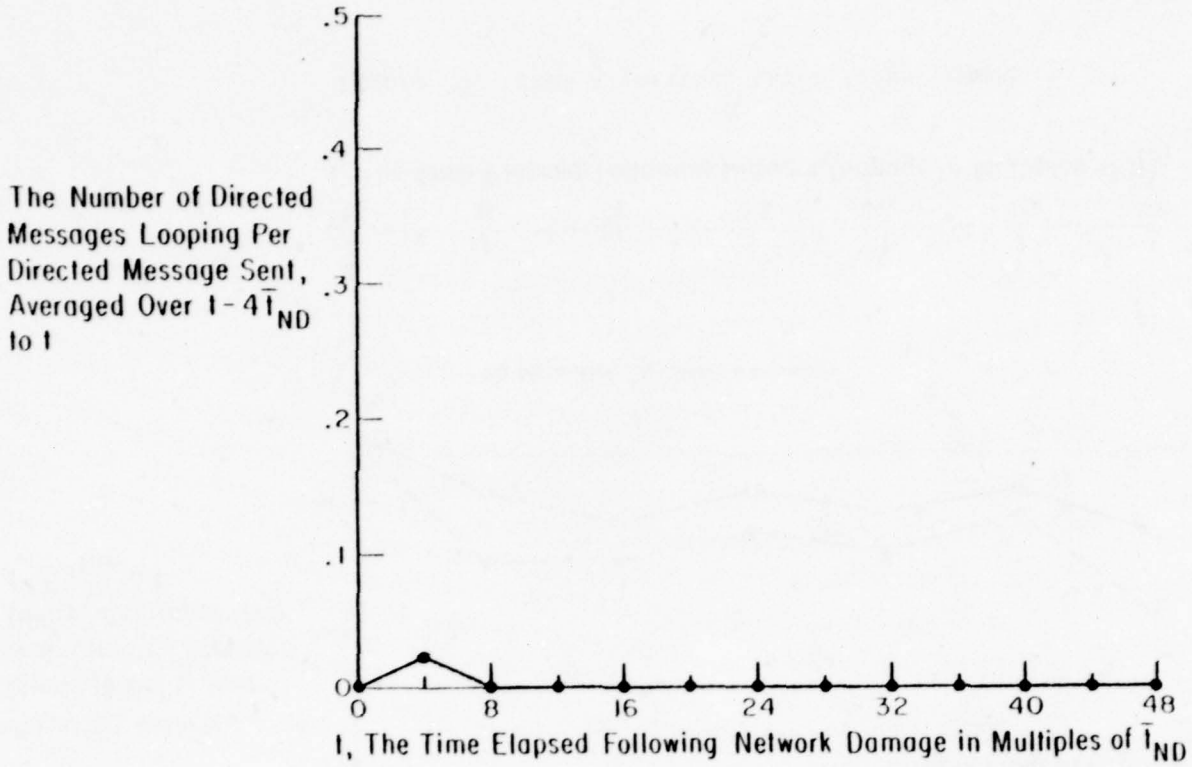


Figure 19. RRPS Algorithm, three links jammed.

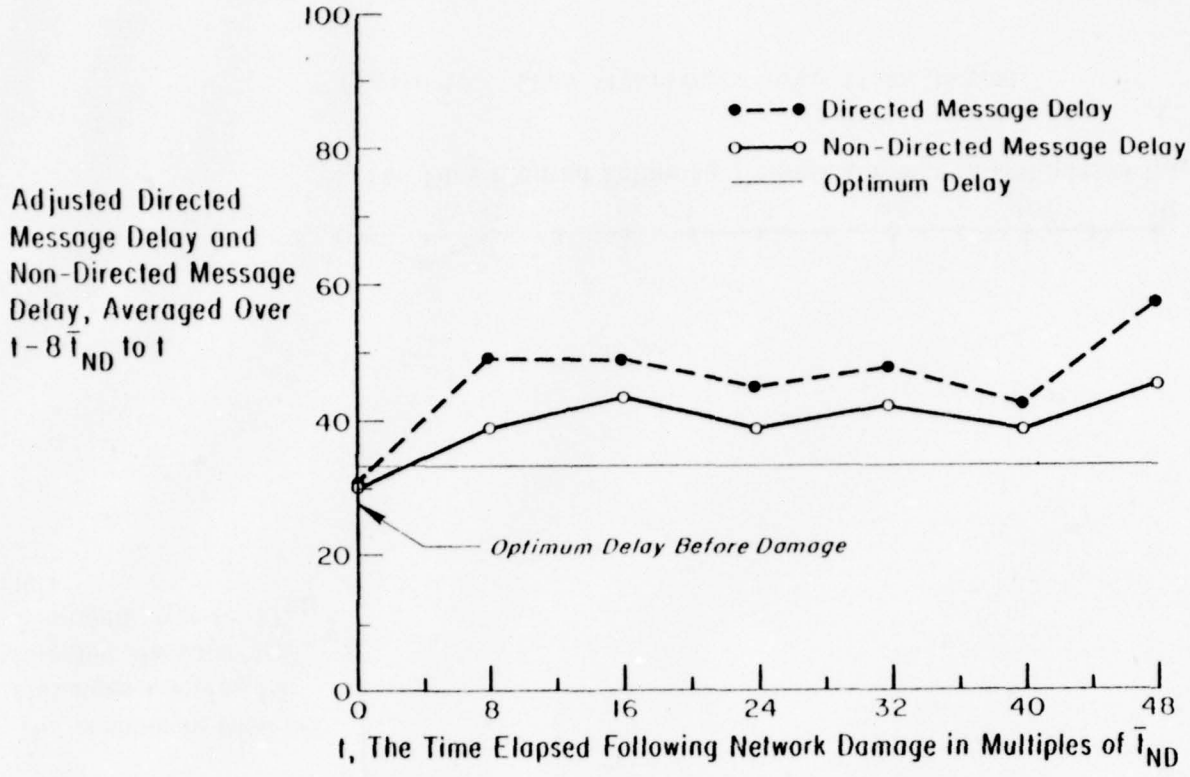


Figure 20. RRPS Algorithm, three links jammed.

about one third as many messages were lost as with the Reciprocal Path Search algorithm.

The adjusted directed message delay shows a gradual rise from its pre-damage value. No large initial jump, such as that observed with the Reciprocal Path Search algorithm, occurs since so few messages are lost. After the routing tables have adapted, and messages are no longer being lost, the directed message delay remains larger than the non-directed message delay. This indicates that some of the directed messages are being routed over sub-optimal paths.

4.2.4 Reciprocal Path Search Algorithm with Delay Vectors

This algorithm sends out the delay vectors at regular intervals spaced T_{DV} apart. The results presented here use $T_{DV} = 4\bar{t}_{ND}$, but the case $T_{DV} = 2\bar{t}_{ND}$ is also treated in the detailed report.

The results for the three links jammed case using the Reciprocal Path Search Algorithm with delay vectors are given in Figures 21, 22, and 23. These results again show a temporary period when messages are lost and looping. This period lasts for about $12\bar{t}_{ND}$. When compared to the Reciprocal Path Search algorithm (without delay vectors), it is clear that the use of delay vectors results in an increase in lost and looping messages.

The results for the directed message delay again show the initial large value due to lost messages. Following this initial period, however, the directed message delay compares favorably with the non-directed message delay. This indicates that the directed messages are being routed over shortest paths. This is an improvement over the corresponding results with the Reciprocal Path Search algorithm. The Reciprocal Path Search

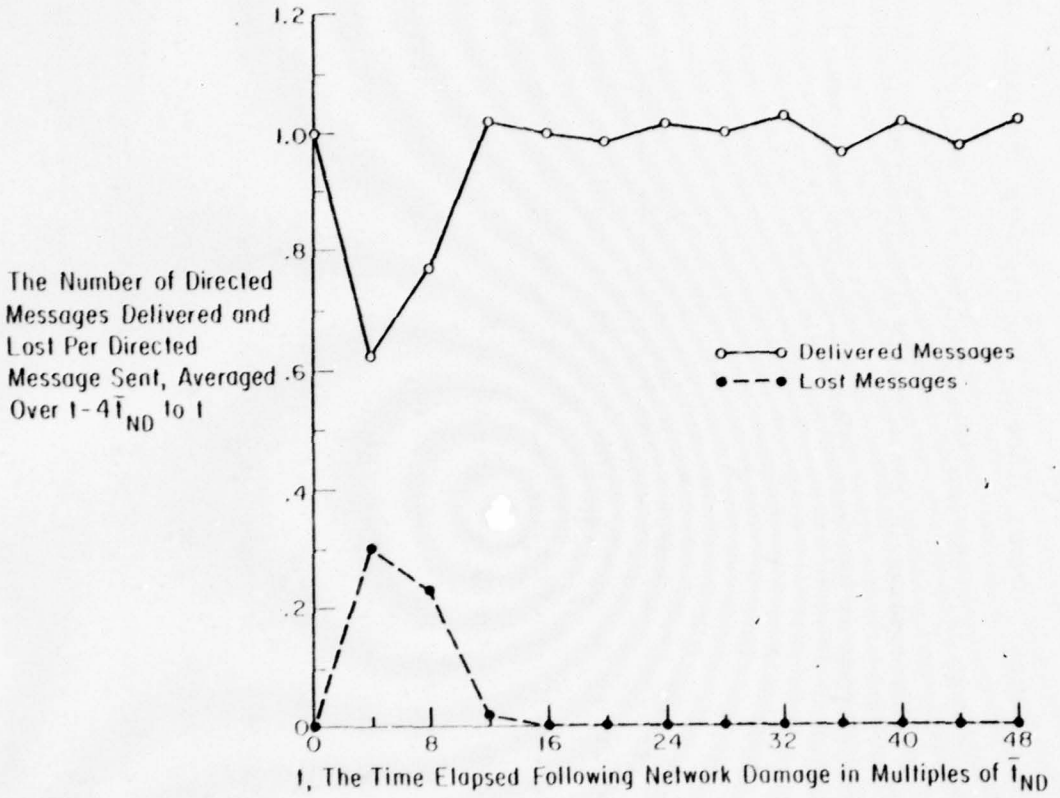


Figure 21. RPSDV Algorithm, three links jammed.

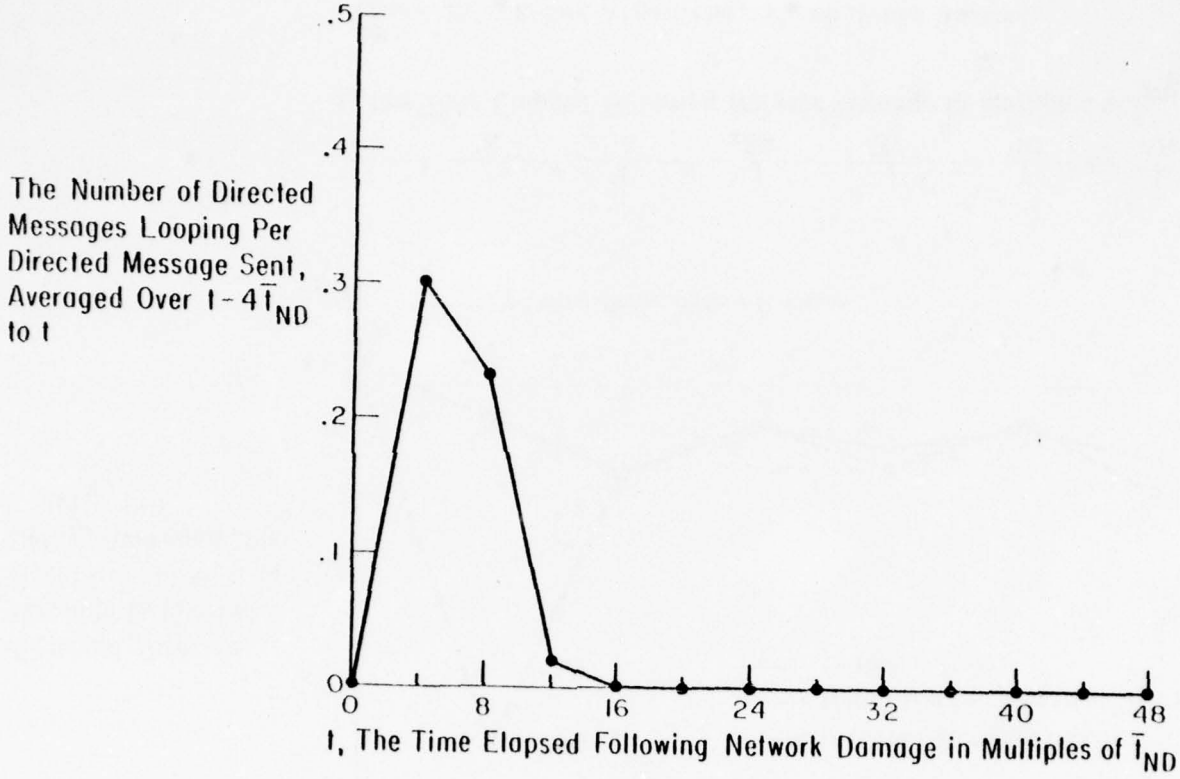


Figure 22. RPSDV Algorithm, three links jammed.

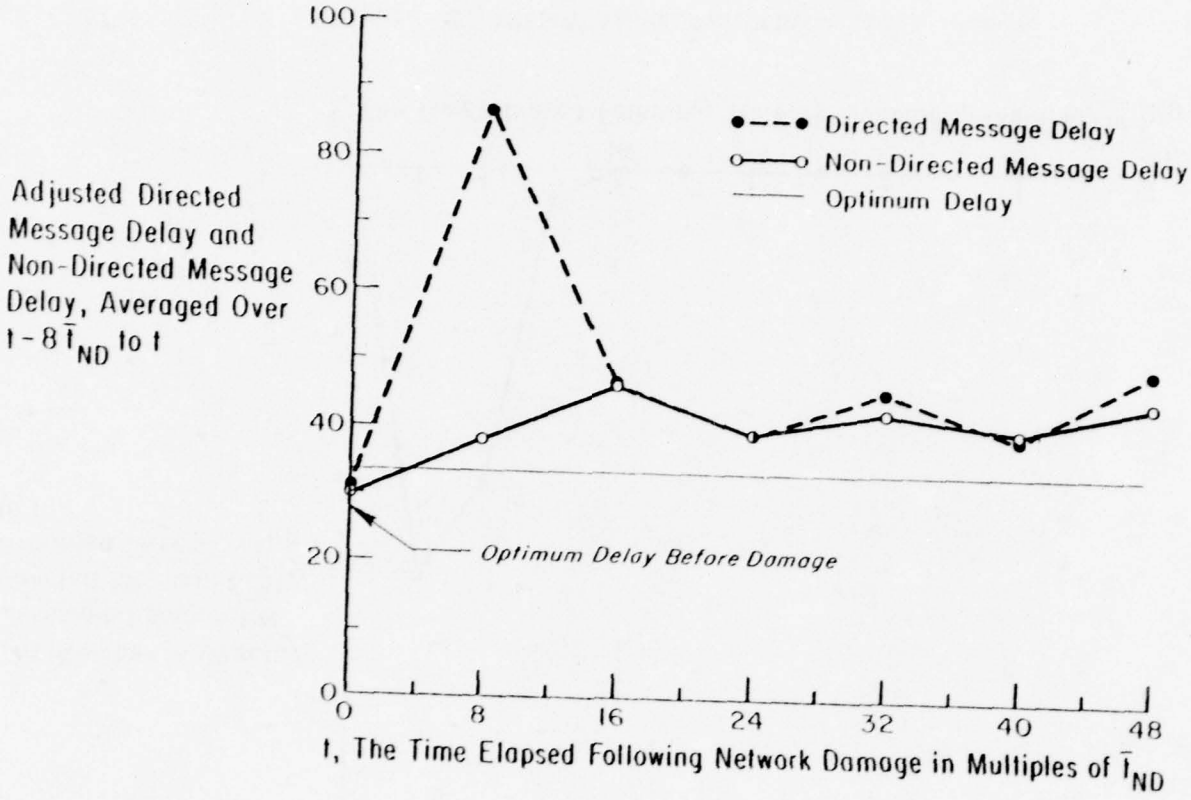


Figure 23. RPSDV Algorithm, three links jammed.

algorithm did not use delay vectors and routed messages only over reciprocal paths, resulting in some messages taking longer than necessary routes. The use of delay vectors enables the directed messages to be routed over non-reciprocal links and thus take the shortest routes to the destination nodes. Thus, although the use of the delay vectors causes an early increase in the number of lost messages, it also allows the messages to take improved routes and thus decrease the steady state delay.

4.2.5 Rapid Reciprocal Path Search Algorithm with Delay Vectors

As with the RPSDV algorithm, the results presented here use $T_{DV} = 4\bar{t}_{ND}$.

The results for the three links jammed case using the Rapid Reciprocal Path Search algorithm with Delay Vectors are given in Figures 24, 25, and 26. Immediately following network damage, the algorithm initially appears to adapt very quickly. However, it had some difficulty completely adapting, which is evidenced by the additional messages being lost at $t = 32\bar{t}_{ND}$ and $44\bar{t}_{ND}$. Even with this trouble in completely adapting, the algorithm performed much better for this case of damage than did the Reciprocal Path Search algorithm with Delay Vectors, although it did lose a few more messages than did the Rapid Reciprocal Path Search algorithm.

The adjusted directed message delay shows a gradual rise from its pre-damage value. Again, there is no large initial jump since so few messages were lost. The adjusted directed message delay then settles out to a value which is generally larger than the non-directed message delay, indicating that the Delay Vector Table restrictions are preventing some of the directed messages from taking shortest routes to their destinations. Although messages are still being lost after the initial adjustment period, so few are that

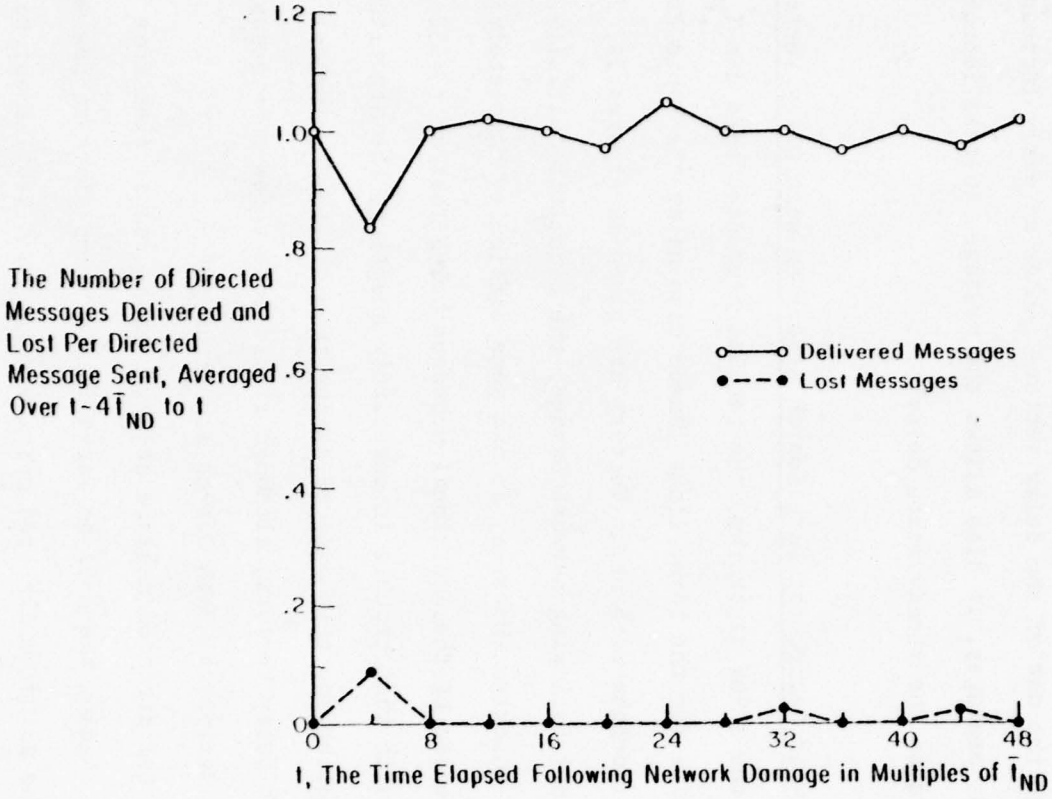


Figure 24. RRPSDV Algorithm, three links jammed.

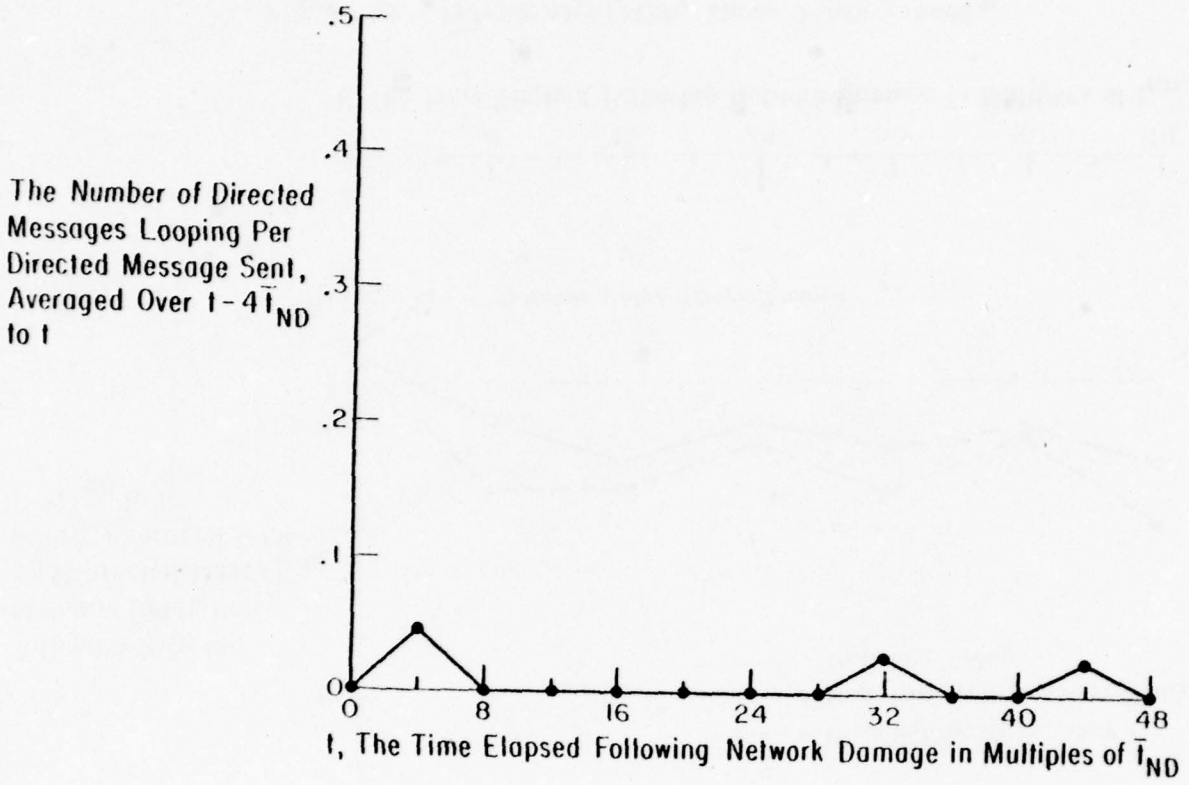


Figure 25. RRPSDV Algorithm, three links jammed.

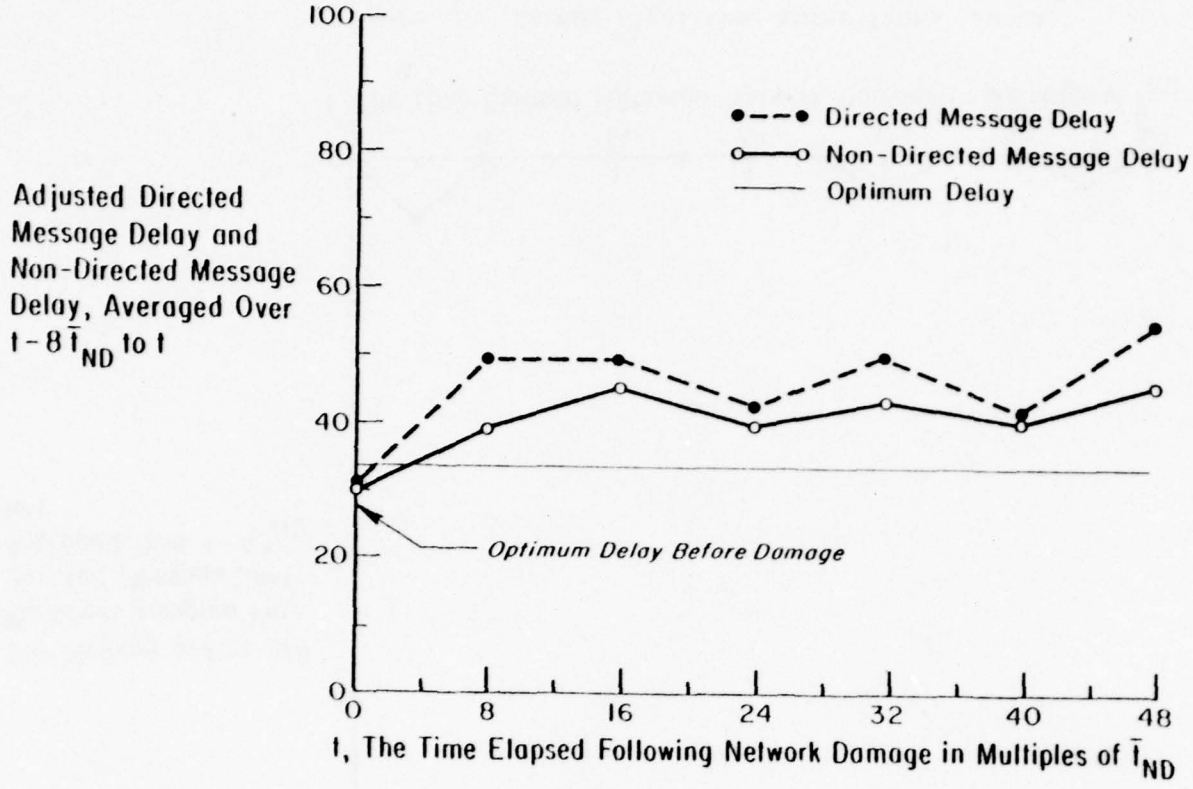


Figure 26. RRPSDV Algorithm, three links jammed.

their effect on the adjusted directed message delay is not noticeable.

4.3 Six Links Jammed Case

4.3.1 Simple Backwards Learning

Figures 27 and 28 show the simulation results for the six links jammed case using the Simple Backwards Learning algorithm. As in the three links jammed case, large numbers of messages are lost and looping through the duration of the test. Many more messages are lost and looping than with the three links jammed case, which is expected since the damage to the network is much more severe.

4.3.2 Reciprocal Path Search Algorithm

The results for the six links jammed case are shown in Figures 29 and 30. Unlike the three links jammed case where all node pairs have at least one reciprocal path between them, the six links jammed case has several node pairs with only non-reciprocal paths between them. Since the Reciprocal Path Search algorithm depends upon the existence of a reciprocal path between nodes, it is unable to adapt to this case of damage.

4.3.3 Rapid Reciprocal Path Search Algorithm

The results for the six links jammed case are contained in Figures 31, 32, and 33. In contrast to the Simple Backwards Learning and Reciprocal Path Search algorithms, the Rapid Reciprocal Path Search algorithm adapted quickly and completely to this severe case of jamming. Messages were lost for only a period of $8\bar{t}_{ND}$. The algorithm was able to quickly locate the new reciprocal routes, and to determine when no reicprocal path exists between

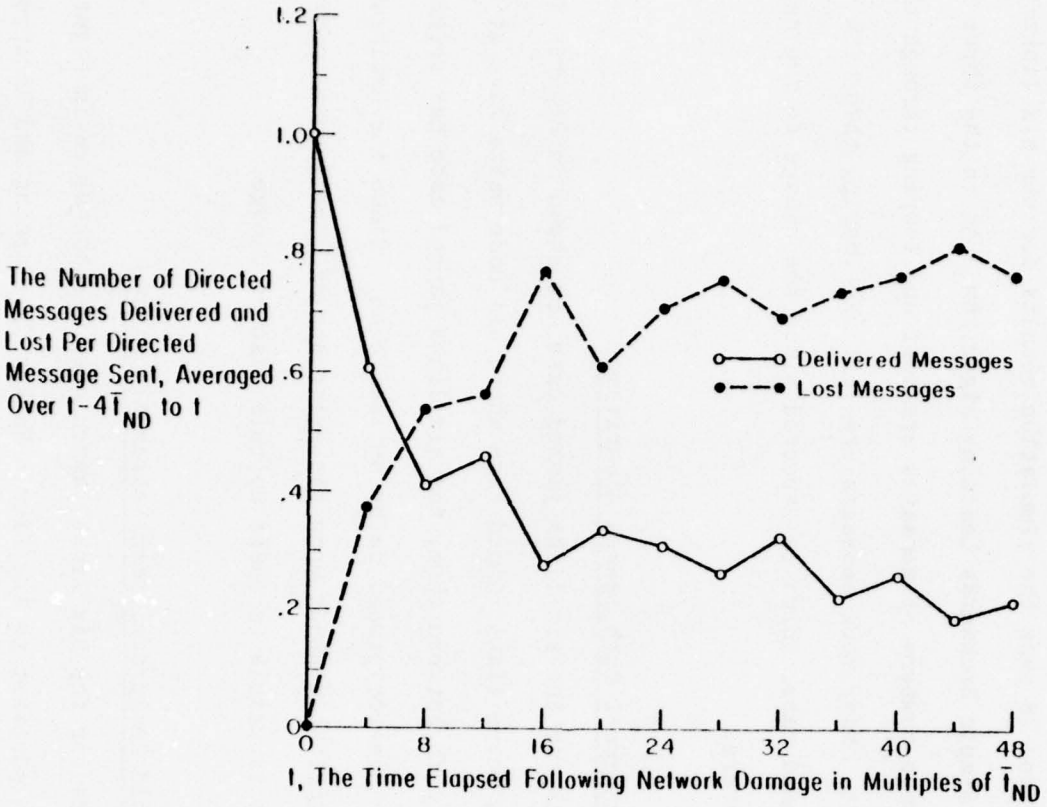


Figure 27. SBL Algorithm, six links jammed.

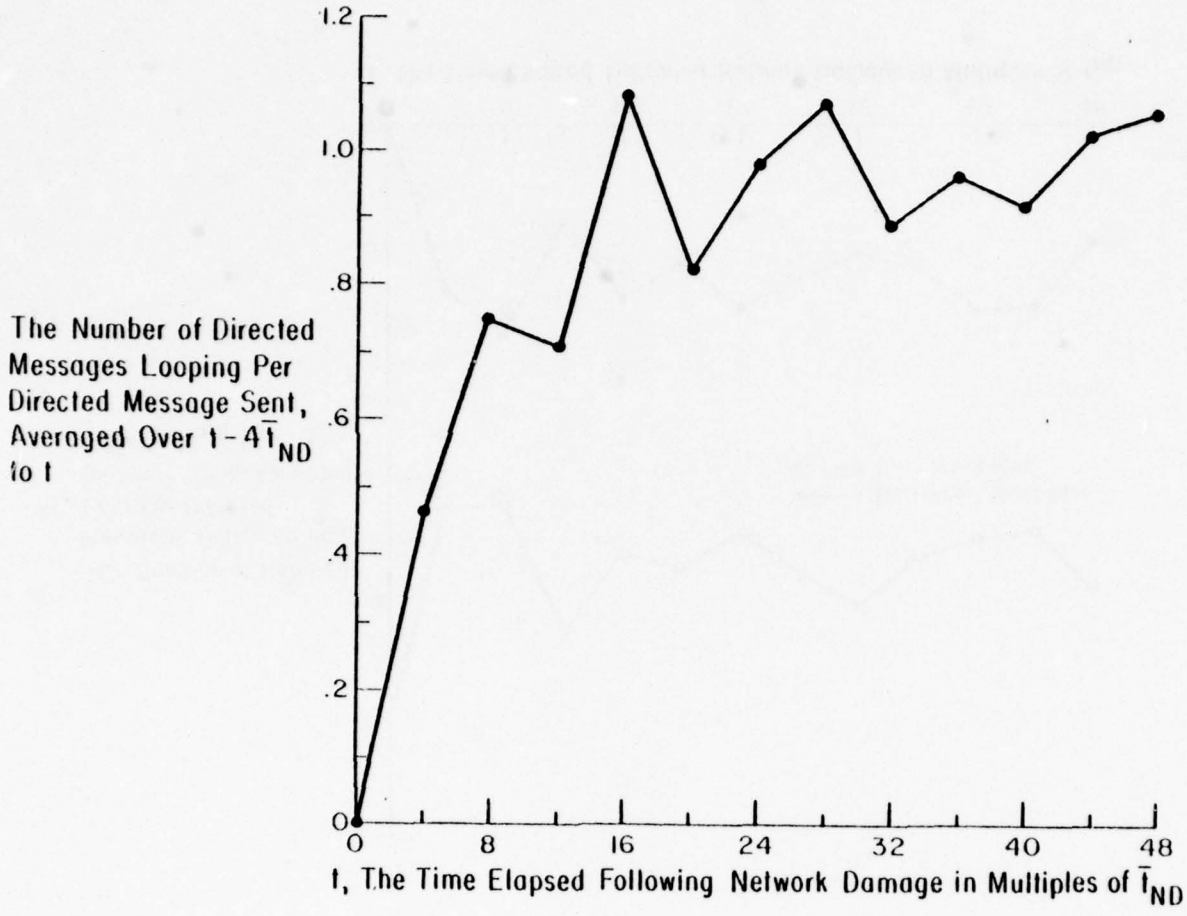


Figure 28. SBL Algorithm, six links jammed.

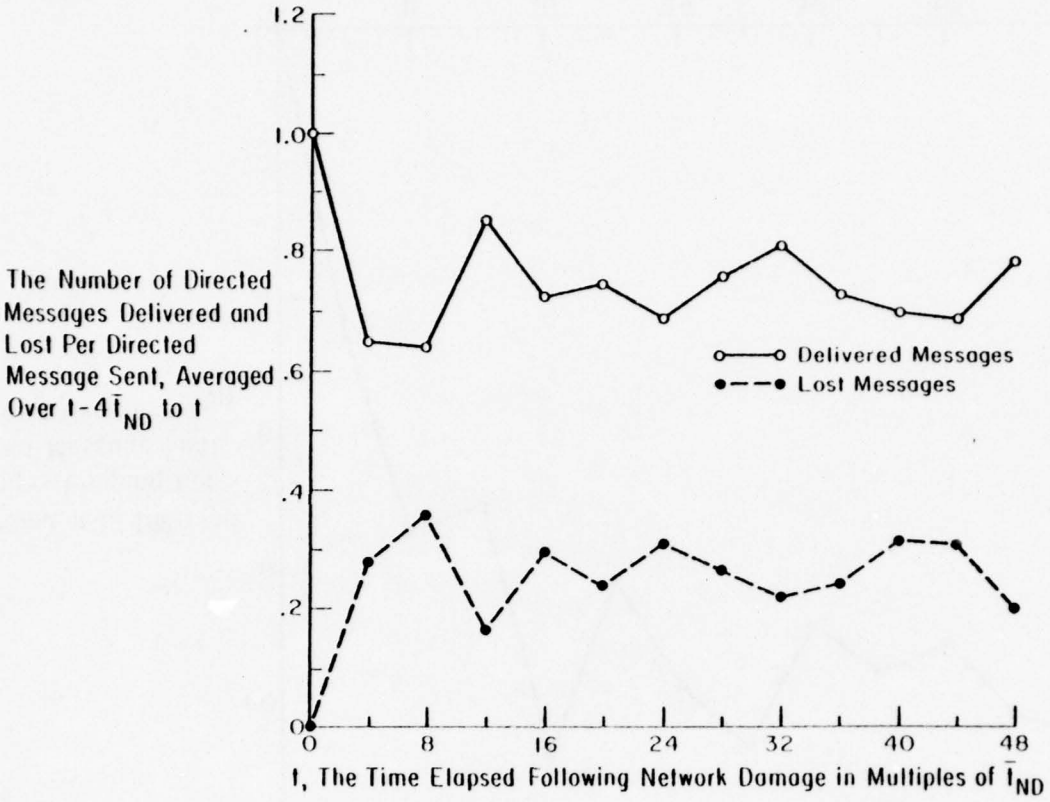


Figure 29. RPS Algorithm, six links jammed.

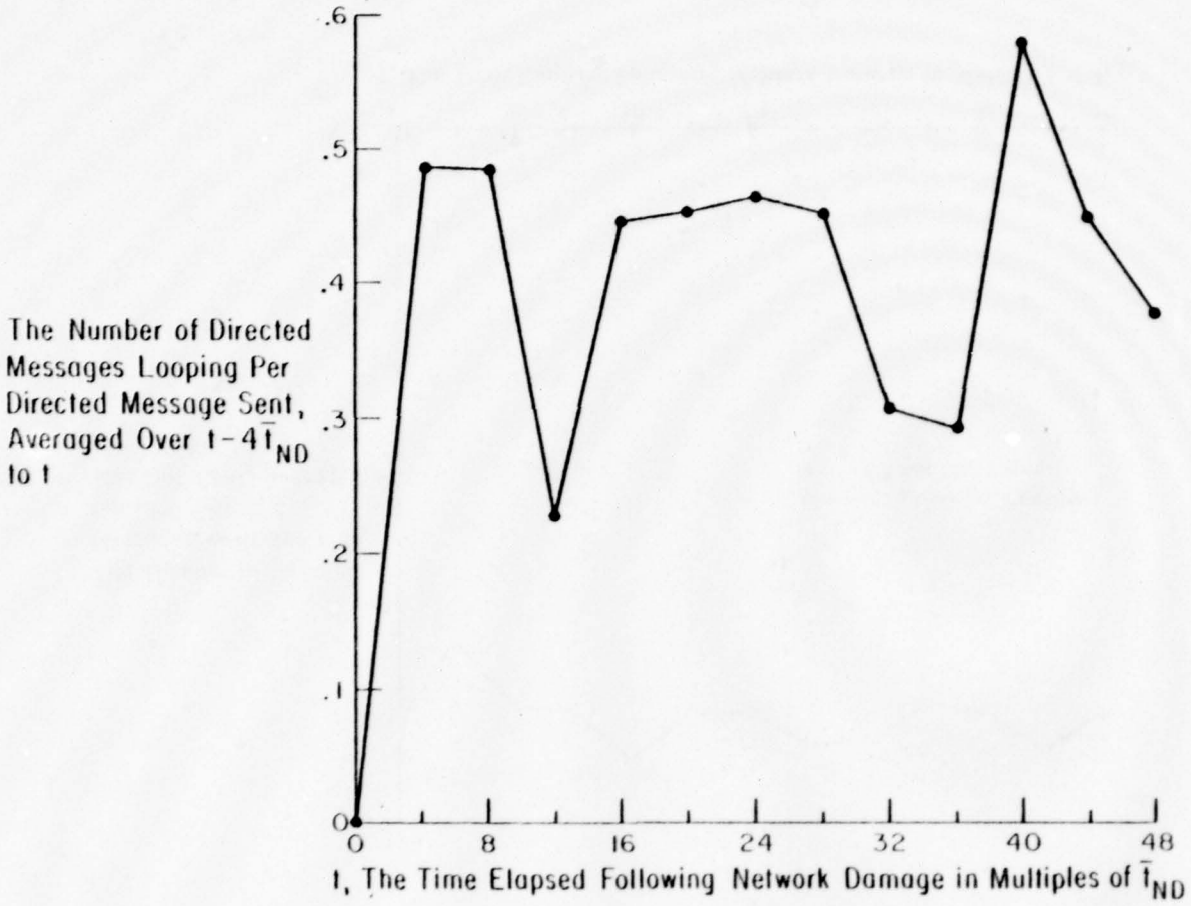


Figure 30. RPS Algorithm, six links jammed.

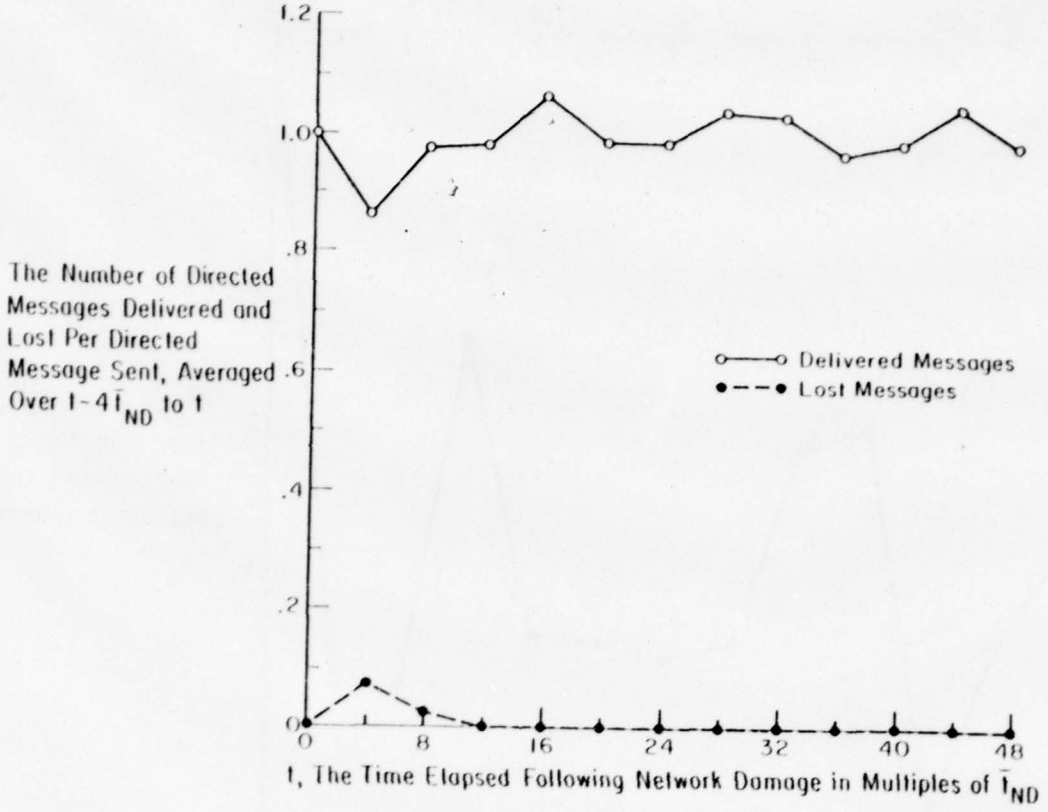


Figure 31. RRPS Algorithm, six links jammed.

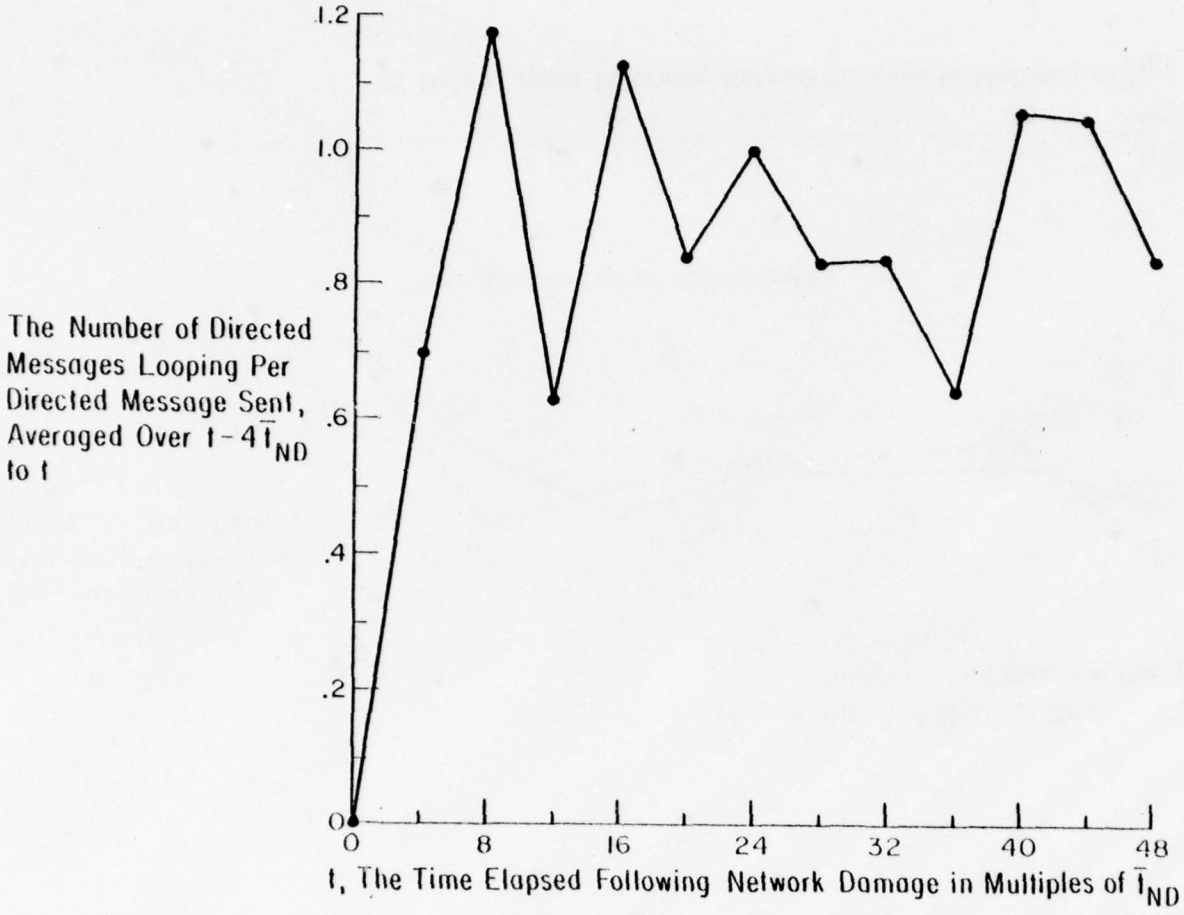


Figure 32. RRPS Algorithm, six links jammed.

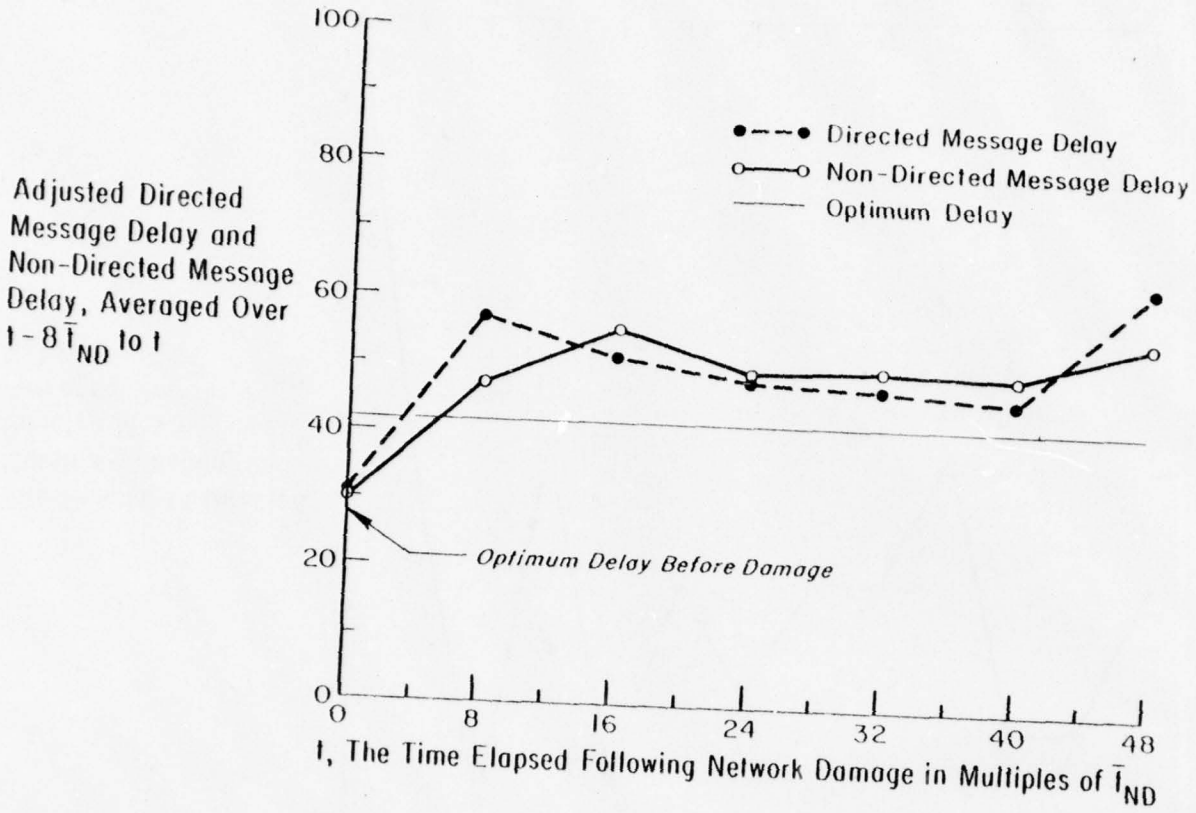


Figure 33. RRPS Algorithm, six links jammed.

nodes. When no reciprocal path is found, the algorithm simply sends the message out of all the node's ports, i.e., it floods the message out. One side effect of this flooding is that a large amount of looping occurs.

The extra traffic caused by the directed message flooding does not seriously contribute to the network congestion since the misdirected copies very quickly ping-pong, and hence are quickly discarded. Once a copy of the message reaches a node which does have a reciprocal path to the destination, it is no longer flooded, but routed out of a single port. Thus, the directed message flooding tends to be local in nature.

The adjusted message delay again shows a gradual increase from its pre-damage value. No large jump is seen since there are so few lost messages. Once the messages are no longer being lost, the directed message delay compares very favorably with the non-directed message delay. This is because the flooding of the directed messages tends to route them over the shortest paths, and, significantly, with this severe case of jamming, there are not a lot of alternate routes, so the possibility of their taking a suboptimal path is low.

4.3.4 Reciprocal Path Search Algorithm with Delay Vectors

Again, the results presented here are for the case $T_{DV} = 4\bar{t}_{ND}$. These results are presented in Figures 34, 35, and 36. These results are a dramatic improvement over the Reciprocal Path Search algorithm, which does not use delay vectors. Without the use of delay vectors to locate the non-reciprocal links, the previous algorithm was unable to adapt to the six links jammed case and messages were lost and looping continuously after the damage occurred. However, the Reciprocal Path Search Algorithm with Delay Vectors

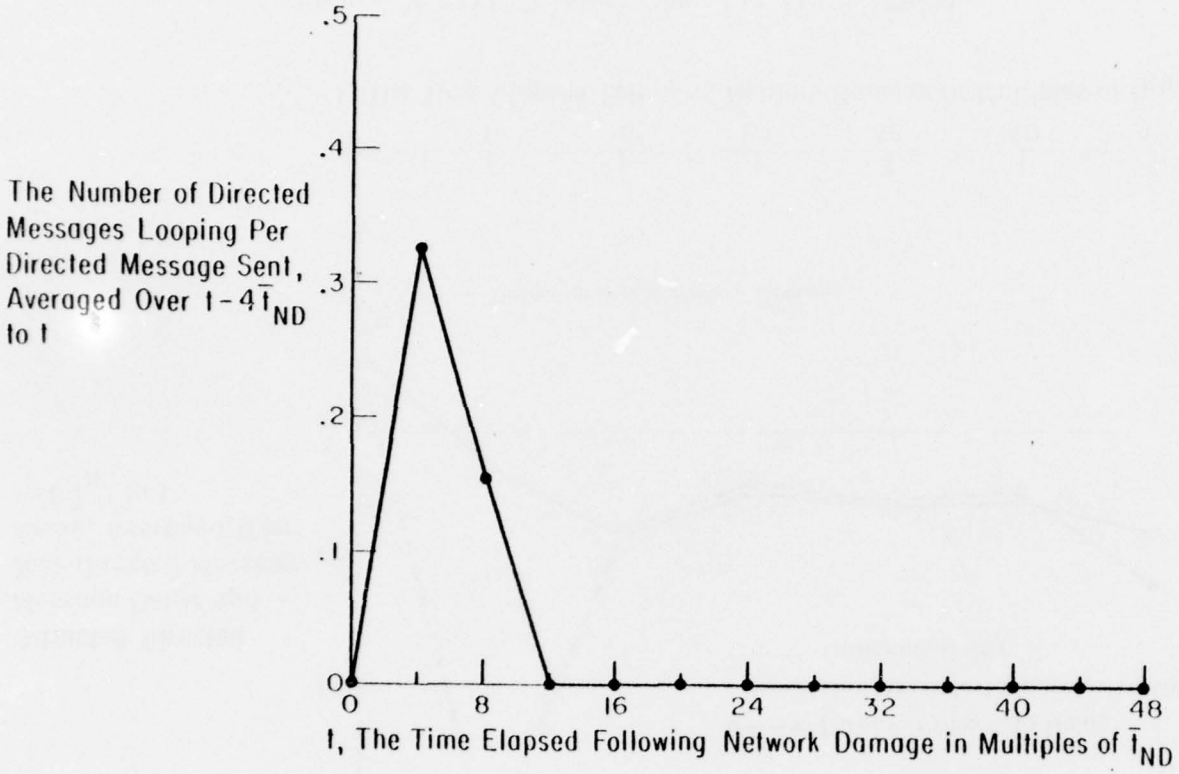


Figure 35. RPSDV Algorithm, six links jammed.

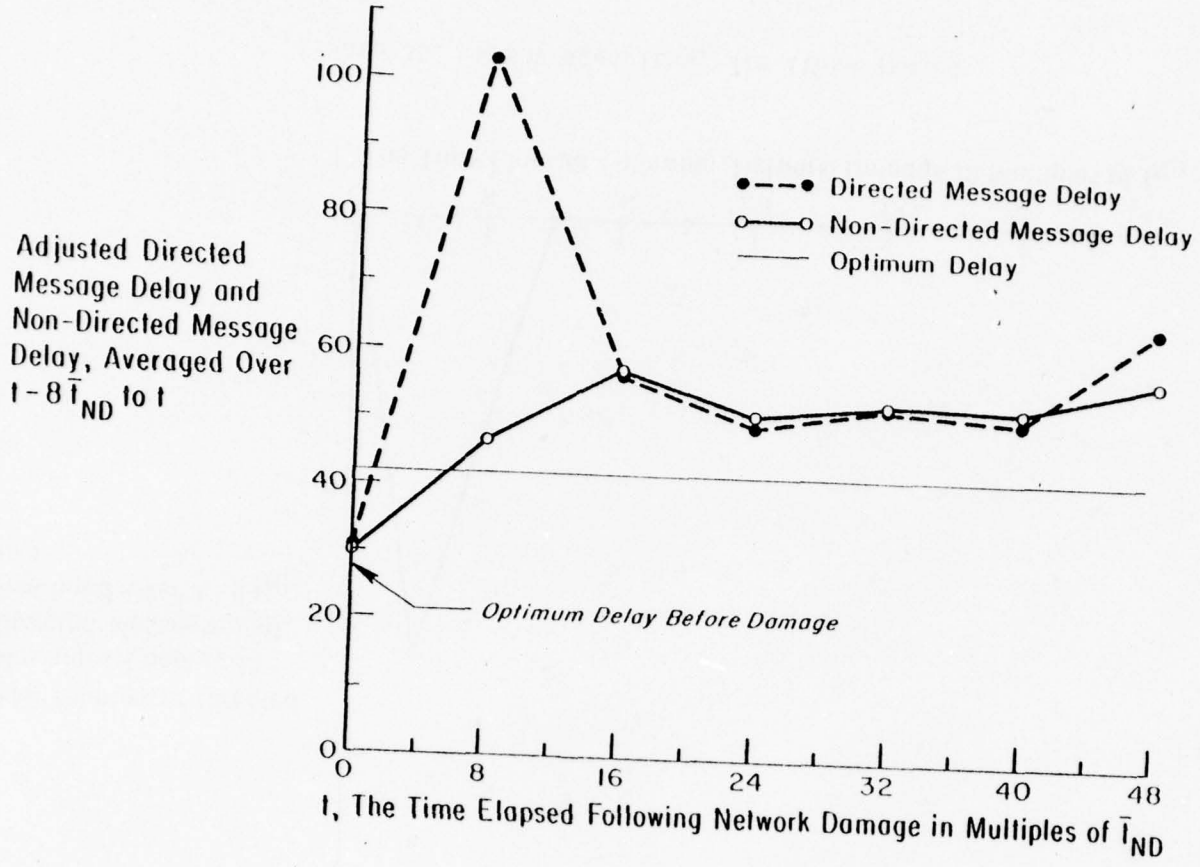


Figure 36. RPSDV Algorithm, six links jammed.

is able to completely adapt to this case of network damage.

Following an initial period of large delay due to lost messages, the directed message delay becomes essentially equal to the non-directed message delay. This indicates that the Reciprocal Path Search Algorithm with Delay Vectors is able to route the messages over the shortest paths.

In summary, it is evident that for severe cases of network damage, the Reciprocal Path Search Algorithm with Delay Vectors is a definite improvement over the Reciprocal Path Search algorithm, which does not use delay vectors. Since the use of delay vectors is necessary for the Reciprocal Path Search algorithm to completely adapt to severe cases of network jamming, they are an essential addition to the algorithm.

4.3.5 Rapid Reciprocal Path Search Algorithm with Delay Vectors

Again, the results presented here are for the case $T_{DV} = 4\bar{t}_{ND}$. These results are presented in Figures 37, 38, and 39. They show an initial period when messages are lost and looping. A few messages are lost even after the looping has ceased, and even while messages are looping many more are being lost. This is a direct result of the algorithm preventing directed messages from being flooded out. Since this algorithm has the capability to route messages over both reciprocal and non-reciprocal paths, the node assumes that if information on a path is not available then no path exists. This is a valid conclusion in steady state and provides the nodes with the capability to detect when they have become disconnected from other network nodes, and thus prevents directed messages which have no hope of reaching their destination from being transmitted and needlessly congesting the network. It is conceivable that, in some situations, this capability can pre-

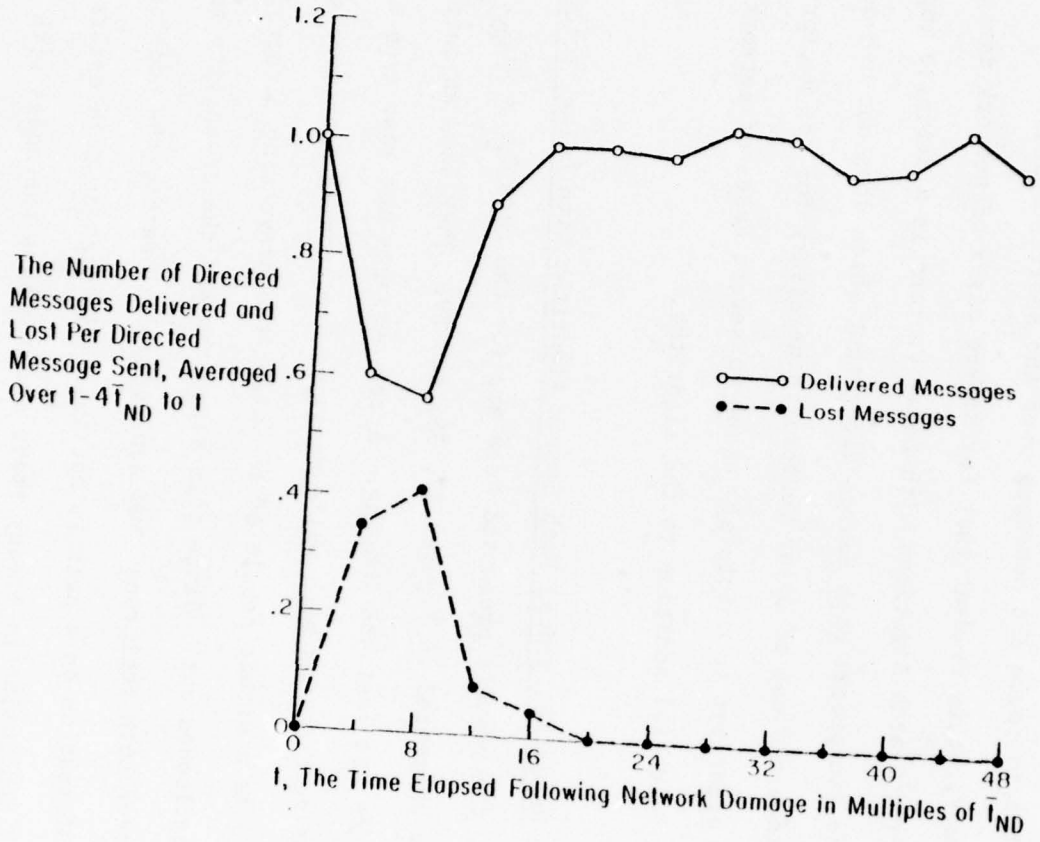


Figure 37. RRPSDV Algorithm, six links jammed.

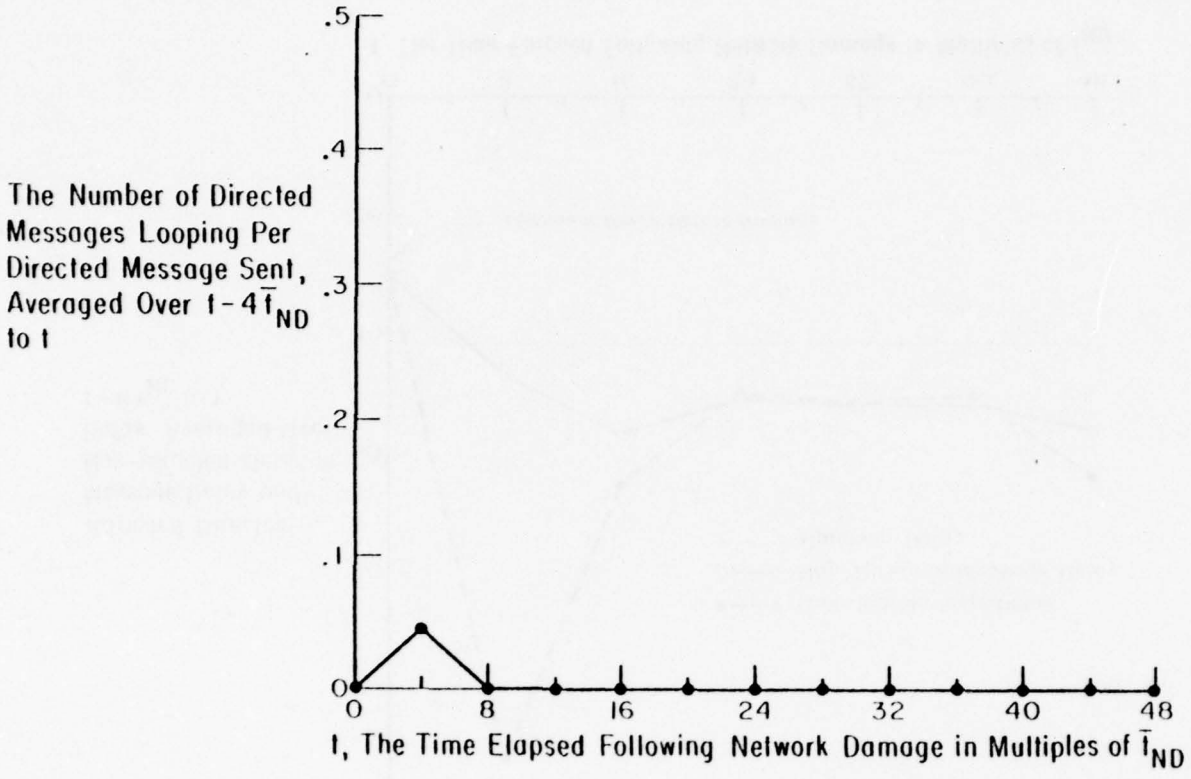


Figure 38. RRPSDV Algorithm, six links jammed.

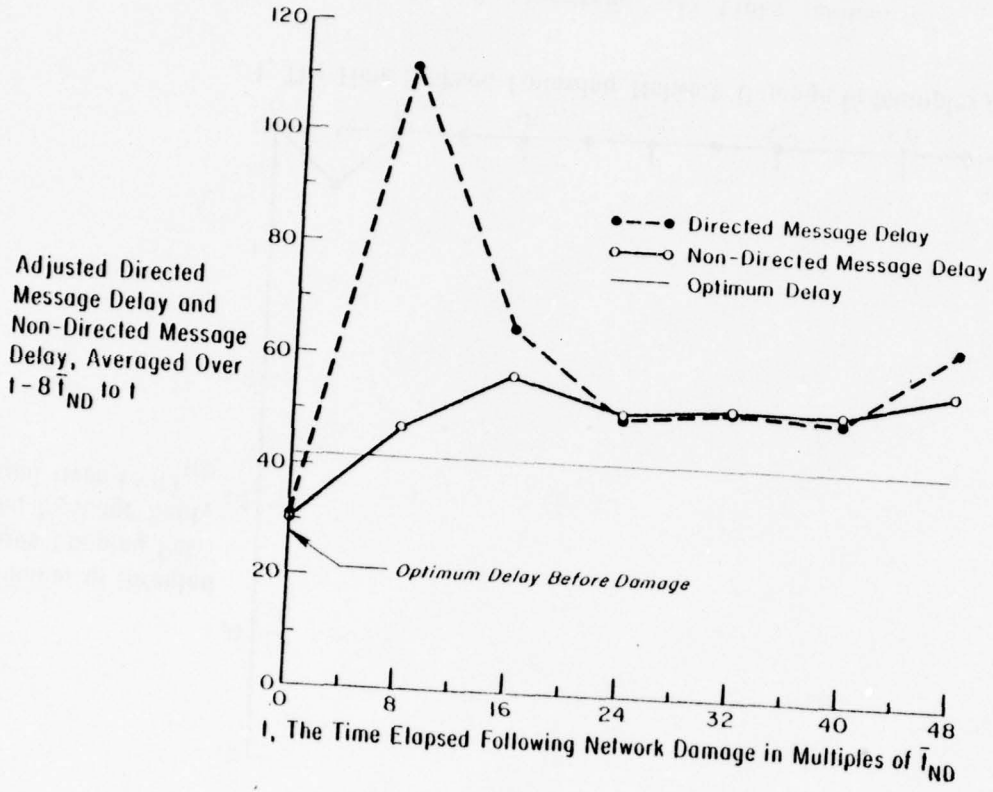


Figure 39. RRPSDV Algorithm, six links jammed.

vent the network from being saturated with directed messages which should not be transmitted, and thus allow the network to continue functioning when it otherwise wouldn't. This feature also slows down the algorithm's adaptation since it takes time for information on the non-reciprocal paths to reach all nodes.

The adjusted directed message delay shows an initial period when delay is much larger than the non-directed message delay due to the lost messages. After the algorithm adjusts, the directed message delay is essentially equal to the non-directed delay indicating the directed messages are being routed over shortest paths.

5. Combined Routing and Acknowledgement Protocols

5.1 Reciprocal Path Search Algorithm with Delay Vectors and Acknowledgements

End-to-end acknowledgements with retransmission of directed messages after a time out period of T_{PA} were added to the Reciprocal Path Search algorithm with Delay Vectors. The acknowledgements are assumed to fit into a packet of length ℓ_A which is one tenth of ℓ_M, the packet length for regular non-directed and directed messages. The acknowledgements are given priority over all other message types so that they can quickly reach the directed message source nodes, and thus prevent excessive congestion caused by unnecessary retransmissions.

Transient simulation tests were conducted on the combined acknowledgement and routing protocol to determine its adaptability to the case of six links jammed. The statistics used to evaluate the protocol's performance are as follows:

1. The number of directed messages delivered per directed message originally sent.
2. The number of directed message copies (i.e., retransmissions) sent per directed message originally sent.
3. The number of directed messages looping per directed message originally sent.
4. The directed message and non-directed message delays.

Statistics 1, 2, and 3 are averaged over a window of width $4\bar{t}_{ND}$ and statistic 4 is averaged over a window of width $8\bar{t}_{ND}$. Thus any point on the curves of these statistics provides an indication of the near term performance of the protocol.

For these tests, the GPSS simulation model of the network was improved by adding noise to the communication links and by more accurately modeling the jammed links. Noise was added to the communication links by decreasing the probability of a successful transmission over a good communication link from one hundred percent to ninety five percent. Improvement in modeling the jammed links was accomplished by allowing fifty percent of the messages transmitted across them to be successfully received. It is expected that even when a link is being jammed, a significant fraction of the messages will be successfully transmitted across it, with the fifty percent fraction used here chosen to severely test the algorithm.

The combined routing and acknowledgement protocols were tested in the six links jammed case with $T_{PA} = 500$. Figures 40, 41, 42, 43, and 44 give the results for this test.

These results show a period immediately following network damage when message delivery is low and messages loop. Once the routing algorithm has

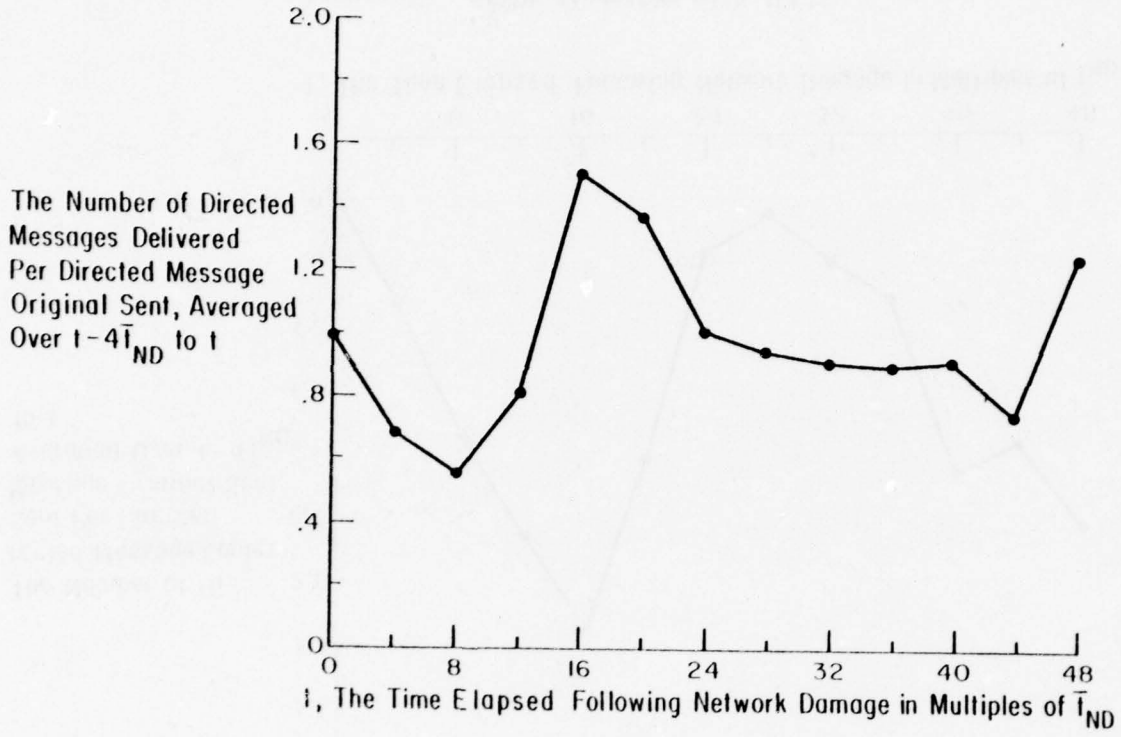


Figure 40. RPSDV Algorithm with ACK's.

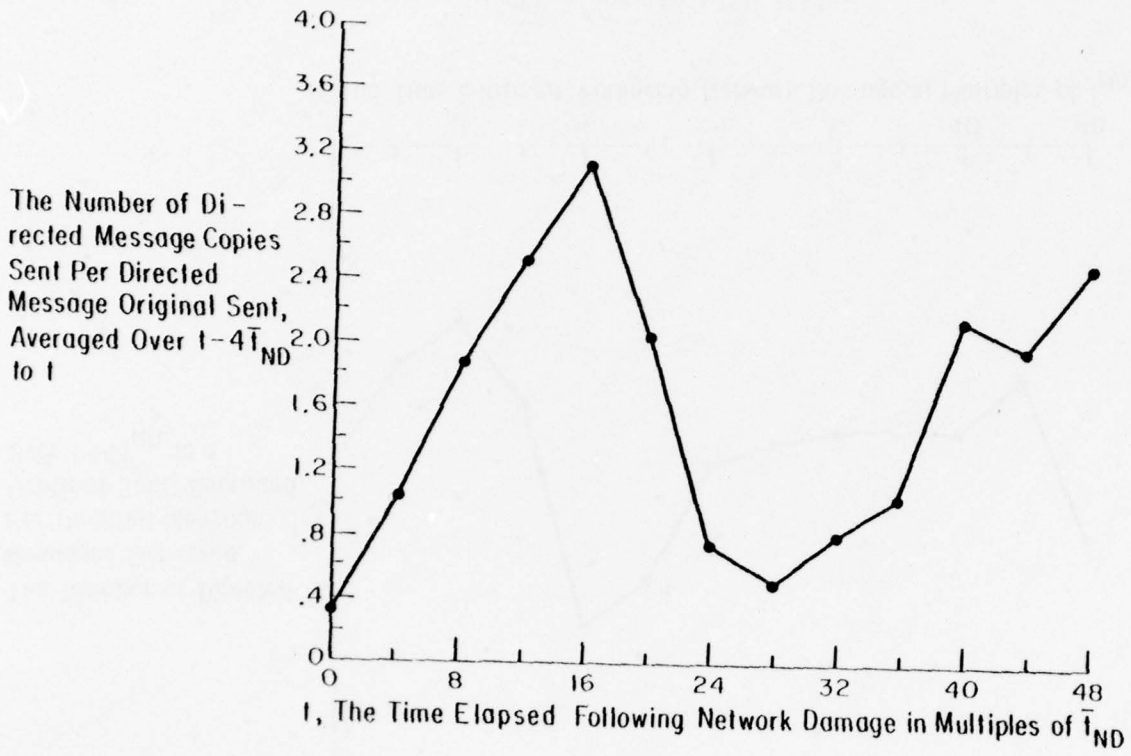


Figure 41. RPSDV Algorithm with ACK's.

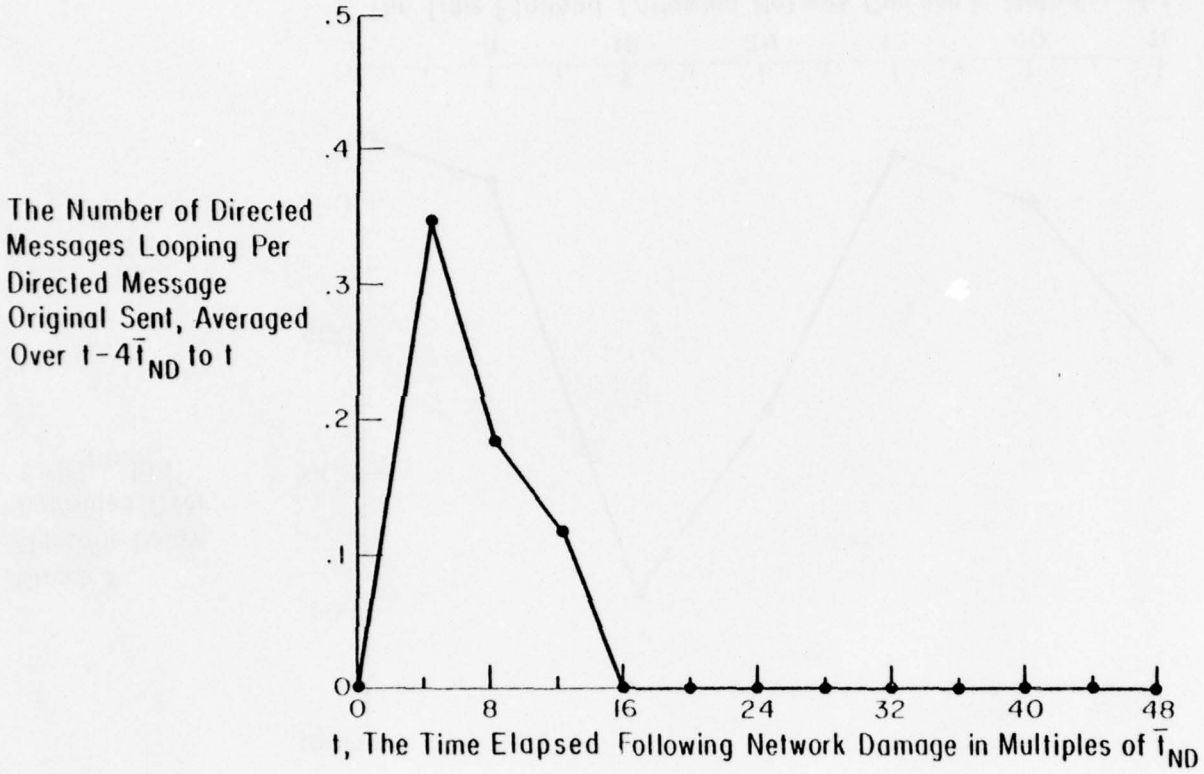


Figure 42. RPSDV Algorithm with ACK's.

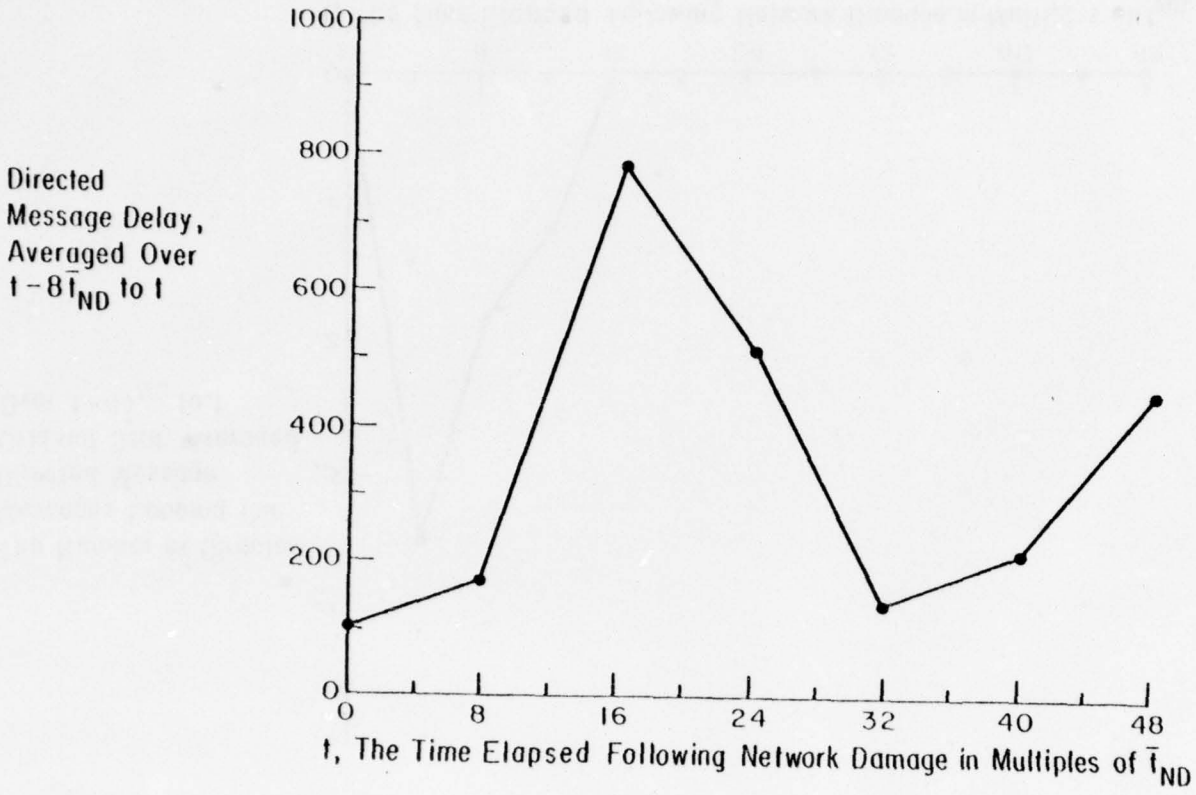


Figure 43. RPSDV Algorithm with ACK's.

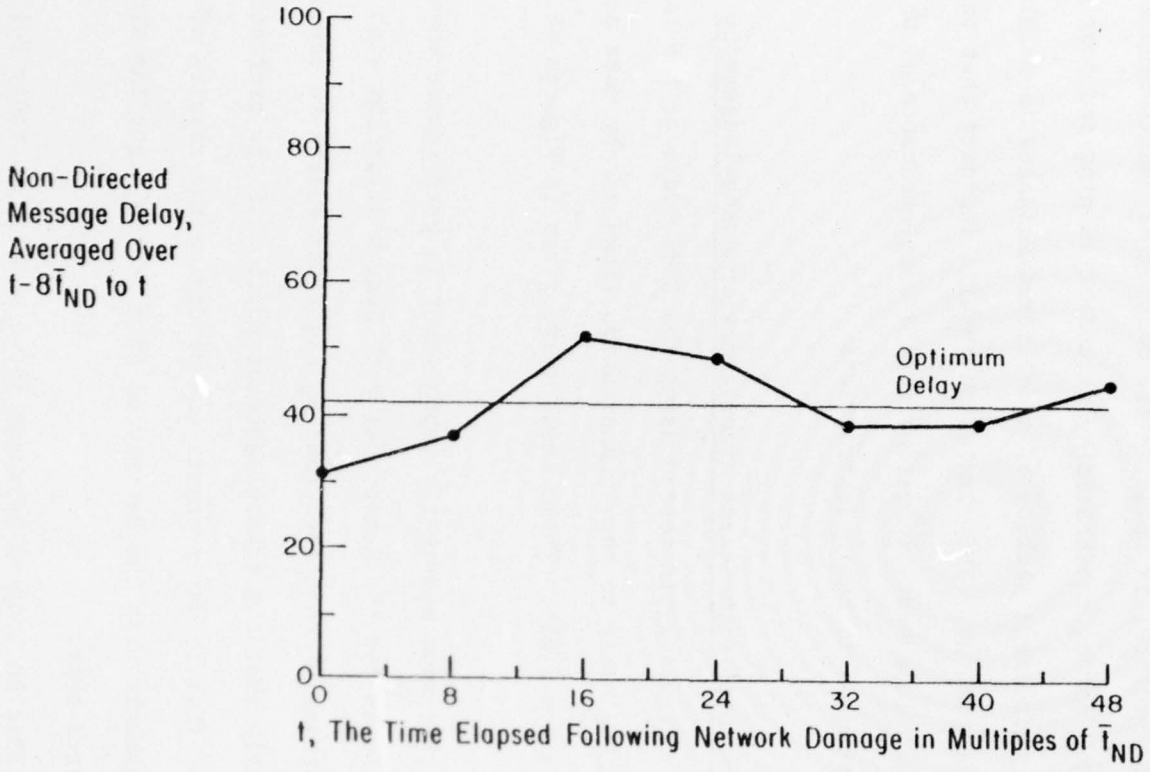


Figure 44. RPSDV Algorithm with ACK's.

adapted to the damage, there is a period of very large message delivery rate. This large delivery rate is a result of large numbers of retransmitted messages finally reaching their destinations. This is a "catch up" period during which the protocol is reducing the backlog of undelivered messages which accumulated during the preceding period of looping and poor message delivery. After this large backlog of undelivered messages is eliminated, the delivery rate levels out. The simulation results indicate that the combined routing and acknowledgement algorithms form a complete protocol which reliably delivers all messages to their destinations.

5.2 Rapid Reciprocal Path Search Algorithm with Acknowledgements

This protocol was again tested using the GPSS model of the radar network modified by adding noise to the communication links and by more accurately modeling the jammed links. These results are given in Figures 45, 46, 47, 48, and 49.

These results show a dramatic improvement in performance when compared to the test results for the Reciprocal Path Search Algorithm with Delay Vectors. The Rapid Reciprocal Path Search routing algorithm adapts to the network damage so quickly that the acknowledgement portion of the protocol is hardly tested at all. This is why emphasis was placed on the results of testing the acknowledgements with the Reciprocal Path Search Algorithm with Delay Vectors discussed above.

The statistic on looping messages shows a large increase following the network damage. However, this is characteristic of the Rapid Reciprocal Path Search routing algorithm.

The statistic on message delivery showed no significant drop following

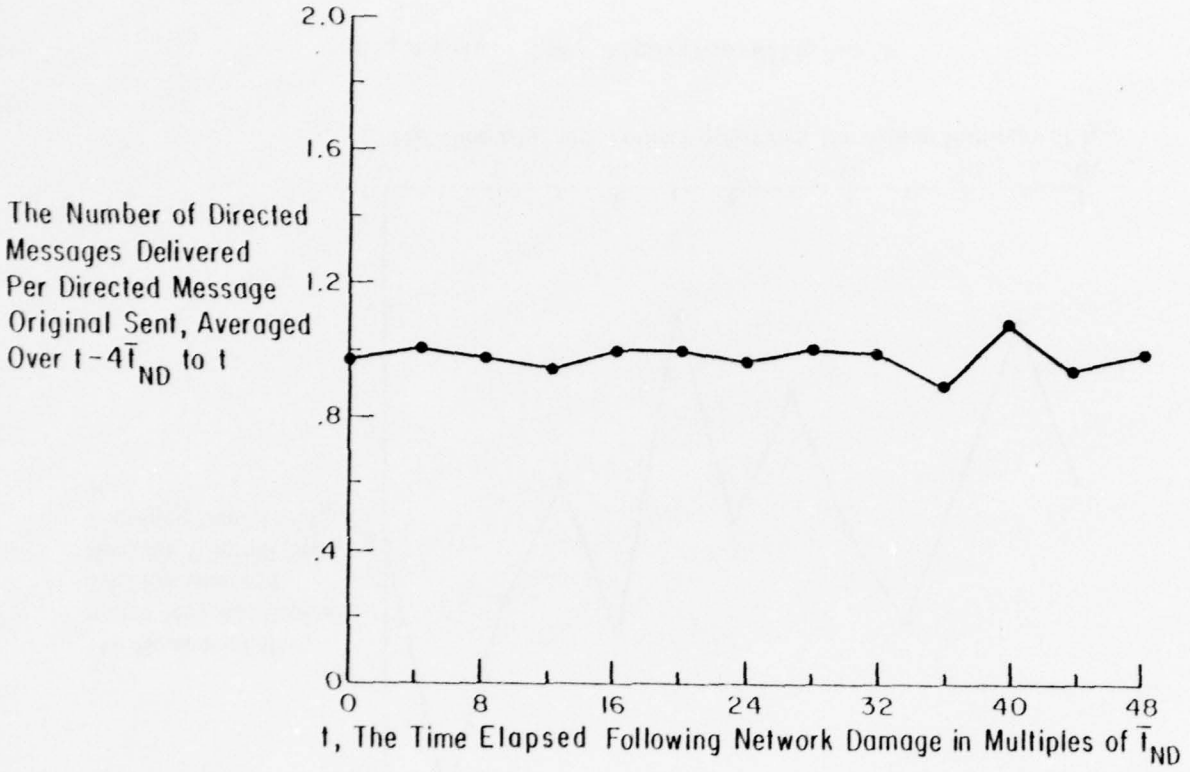


Figure 45. RRPS Algorithm with ACK's.

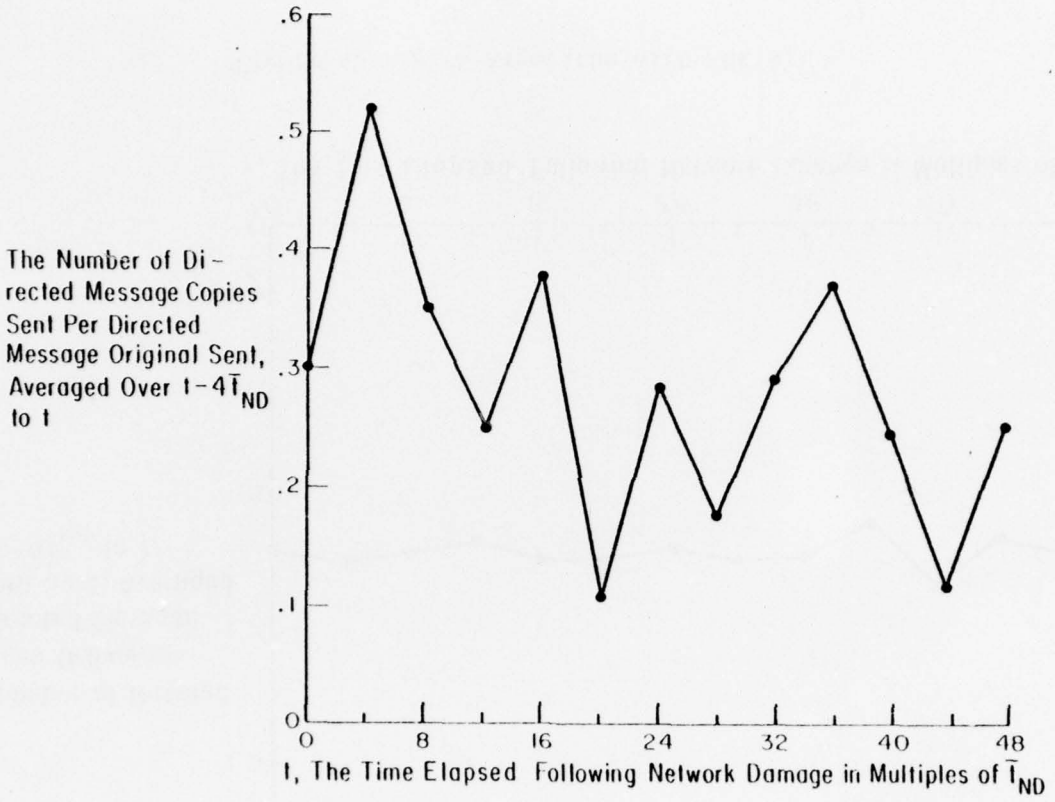


Figure 46. RRPS Algorithm with ACK's.

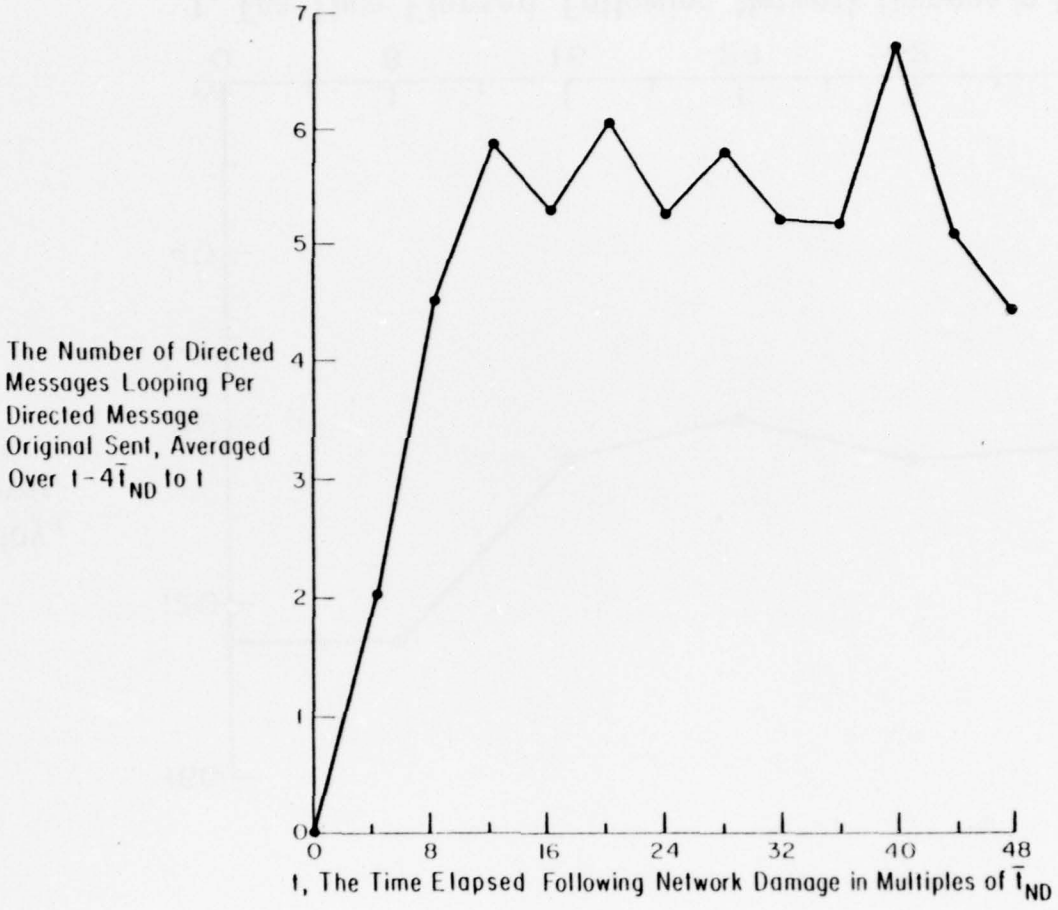


Figure 47. RRPS Algorithm with ACK's.

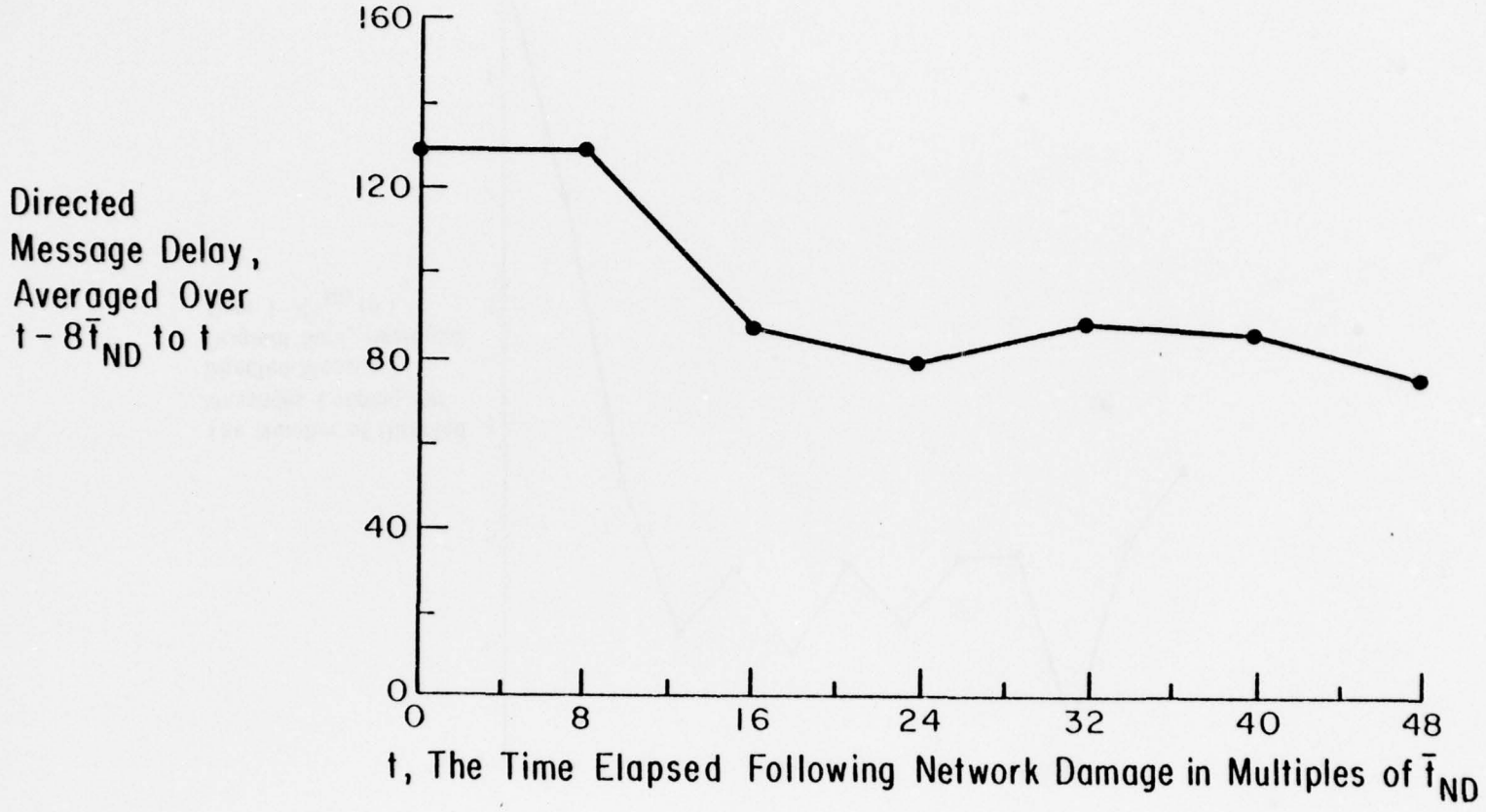


Figure 48. RRPS Alorithm with ACK's.

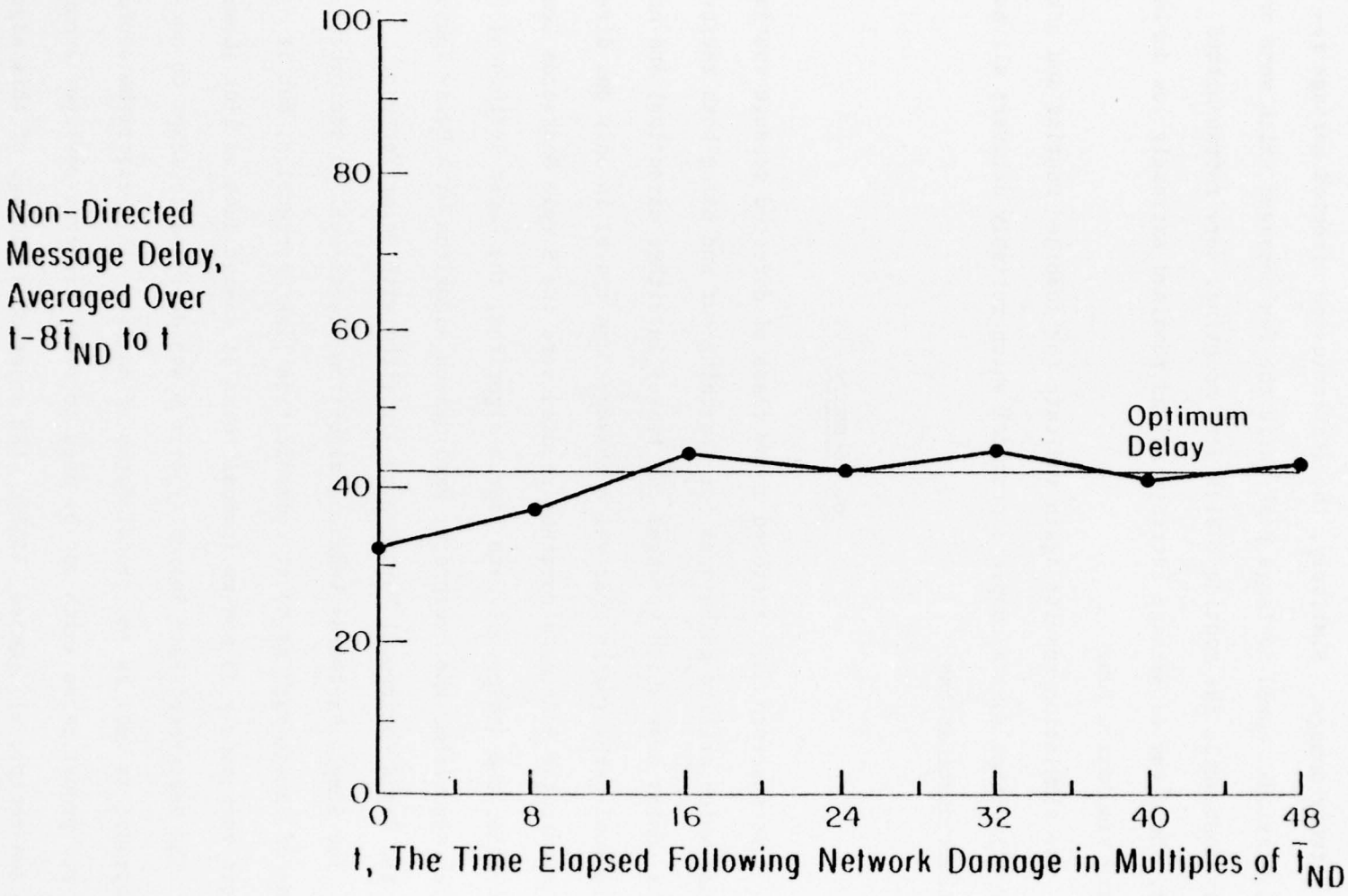


Figure 49. RRPS Algorithm with ACK's.

the network damage. Similarly, the statistics on directed message retransmissions showed no large peak while the few messages which were originally lost while the routing tables were adapting, were retransmitted. In fact, the directed message retransmissions remained extremely low during the entire simulation run.

The simulation results again indicate the combined routing and acknowledgement algorithms form a complete protocol which reliably delivers all messages to their destinations.

6. Summary

This research has developed a new class of directed message routing algorithms which utilize techniques for searching out and using both reciprocal paths (paths over which messages can travel in either direction) and non-reciprocal paths (paths over which messages can travel in only one direction). The particular routing algorithms studied were the Simple Backwards Learning algorithm, the Reciprocal Path Search algorithm, the Rapid Reciprocal Path Search algorithm, the Reciprocal Path Search Algorithm with Delay Vectors, and the Rapid Reciprocal Path Search Algorithm with Delay Vectors.

The Simple Backwards Learning algorithm adapts well to reciprocal forms of damage such as total communication link destruction, but it adapts very poorly to non-reciprocal forms of damage such as link jamming.

The Reciprocal Path Search algorithm was designed to adapt to non-reciprocal as well as reciprocal forms of damage. This algorithm searches out reciprocal paths which can be used to route directed messages around the non-reciprocal (jammed) links. The major disadvantage of this algorithm is that it requires the existence of reciprocal paths between nodes, and

whenever such paths do not exist, the algorithm fails to adapt.

The Rapid Reciprocal Path Search algorithm is designed to adapt more quickly and more completely than does the Reciprocal Path Search algorithm. This algorithm also searches out reciprocal paths which can be used to route directed messages around the non-reciprocal links, and it accomplishes this much more quickly than does the Reciprocal Path Search algorithm. In addition, this algorithm is able to quickly detect the absence of a reciprocal path between nodes and is thus able to deliver directed messages in this situation by flooding them.

The Reciprocal Path Search Algorithm with Delay Vectors is designed to search out and use both reciprocal and non-reciprocal paths which can be used for directed message routing. Whenever any path exists between two nodes, reciprcoal or non-reciprocal, this algorithm is able to locate and use such a path for directed message routing. This algorithm also has the ability to detect the absence of a path between nodes.

The Rapid Reciprocal Path Search Algorithm with Delay Vectors also has the capability to locate and use both reciprocal and non-reciprocal paths for directed message routing, and to detect the absence of any path between nodes. However, it accomplishes these tasks generally more quickly than does the Reciprocal Path Search Algorithm with Delay Vectors.

Two of the routing algorithms, the Rapid Reciprocal Path Search algorithm and the Reciprcoal Path Search Algorithm with Delay Vectors, have been tested in conjunction with an acknowledgement scheme for the case of six links jammed. The test results indicate that both routing algorithms, when combined with acknowledgement schemes, form complete protocols which reliably deliver messages even with severe network jamming.

The results of the adaptability tests show that for the cases of network damage considered in this research, the Rapid Reciprocal Path Search Algorithm clearly performed the best. However, these cases of network damage are necessarily limited in scope, and were chosen only to provide a wide range of conditions under which to test the algorithms. Also, further refinement of the algorithms presented could also alter some of the comparisons made in this research. With further refinements, the Reciprocal Path Search Algorithm with Delay Vectors or the Rapid Reciprocal Path Search Algorithm with Delay Vectors could easily become the preferred algorithm.

7. Areas For Further Research

Performance tests were conducted in this research on each of five new routing algorithms. These tests were limited in scope and were designed primarily to assist in developing the new routing algorithms and to analyze their qualitative performance characteristics while adapting to a wide range of network damage. Among the performance characteristics of most interest were whether or not the routing algorithms are able to adapt, their speed of adaptation if they do adapt, and which of the new routing algorithms are able to route messages over optimal paths. The quantitative results obtained from the transient performance tests are not very meaningful, however, primarily for two reasons. First, the quality of the input data is not good. It is not accurately known what the characteristics of the imput traffic will be in an actual radar network. Second, each of the algorithms was tested with only a single simulation run. Thus, a large amount of confidence should not be placed in the exact numerical quantities obtained in the transient tests.

Since the performance tests made as a part of this research are limited

in scope, much further research needs to be done on the new routing algorithms.

The algorithms have been tested in a thirteen node network. Performance tests should also be conducted using a larger network (a network of fifty or more nodes is the expected size of an actual radar network). In a larger network, it is expected that each routing algorithm will possess the same qualitative characteristics revealed here. However, the size of the network should affect the speed with which each routing algorithm adapts to network damage. A large network will contain both longer paths and more alternate paths than a small network, increasing the amount of time required to disseminate information on reciprocal and non-reciprocal paths throughout the network, thus slowing the routing algorithms' adaptability.

The routing algorithms should also be tested using varying values of k_1, k_2, and k_3 (the learning, forgetting and limiting constants used in updating the routing tables). The values of these parameters, which largely remained fixed throughout the performance tests, have a major effect upon the rate of adaptation of the new routing algorithms to network damage.

The transient tests on the new routing algorithms were also conducted with fixed $\bar{t}_D$ (the mean inter-generation time for directed messages at each node) and $\bar{t}_{ND}$ (the mean inter-generation time for non-directed messages at each node). Further tests should be conducted varying these parameters to determine the performance of the new routing algorithms under different levels of network congestion and varying ratios of directed and non-directed messages.

Tests also need to be conducted on the new routing algorithms using a wide variety of cases of network damage. The adaptability of each new routing algorithm needs to be evaluated using cases of network damage which in-

include destroyed nodes, both destroyed and jammed links together, and which include source-destination node pairs which are completely disconnected from each other.

Finally, many refinements can be made to the routing algorithms. These refinements could greatly improve the performance of the new routing algorithms, particularly the routing algorithms which include delay vectors. Other routing algorithms worth study should also be developed, using the study here as a basis.